

TNO Defence Research

**AD-A273 749**



TNO-report  
FEL-92-A394

copy no.

title

**Zernike Moments and Rotation Invariant Object Recognition  
A Neural Network Oriented Case Study**

author(s):

**P.F.C. Krekel**

date:

**December 1992**

**TDCK RAPPORTENCENTRALE**

Frederikkazerne, gebouw 140  
v/d Burchlaan 31 **MPG 16A**  
TEL. : 070-3166394/6395  
FAX. : (31) 070-3166202  
Postbus 90701  
2509 LS Den Haag **TDCK**

classification

classified by : **M.P.W. van der Weg**

classification date : **September 9, 1993**

title : **Ongerubriceerd**

abstract : **Ongerubriceerd**

report text : **Ongerubriceerd**

appendices A-B : **Ongerubriceerd**

no. of copies : **30**

no. of pages : **87** (including appendices,  
excluding RDP and distribution list)

no. of appendices : **2**

All information which is classified according to Dutch regulations shall be treated by the recipient in the same way as classified information of corresponding value in his own country. No part of this information will be disclosed to any party.

The classification designation **ONGERUBRICEERD** is equivalent to **UNCLASSIFIED**.

All rights reserved.  
No part of this publication may be reproduced and/or published by print, photoprint, microfilm or any other means without the previous written consent of TNO.

In case this report was drafted on instructions, the rights and obligations of contracting parties are subject to either the 'Standard Conditions for Research Instructions given to TNO', or the relevant agreement concluded between the contracting parties.  
Submitting the report for inspection to parties who have a direct interest is permitted.

© TNO

Netherlands organization for  
applied scientific research

TNO Defence Research consists of:  
the TNO Physics and Electronics Laboratory,  
the TNO Prins Maurits Laboratory and the  
TNO Institute for Perception



TNC  
Labx **TD 92-3840**

Oude Waalsdorperweg 63  
2597 AK The Hague  
P.O. Box 96864  
2509 JG The Hague  
The Netherlands

Fax +31 70 328 09 61  
Phone +31 70 326 42 21

①

DEC 15 1993  
S B

**DISTRIBUTION STATEMENT A**  
Approved for public release;  
Distribution Unlimited

**93-30320**



93 12 14 05 4

---

report no. : FEL-92-A394  
title : Zernike Moments and Rotation Invariant Object  
Recognition  
  
A Neural Network Oriented Case study  
  
author(s) : P.F.C. Krekel  
institute : TNO Physics and Electronics Laboratory  
  
date : December 1992  
NDRO no. : A92KL632  
no. in pow '92 : 708  
  
research supervised by : P.F.C. Krekel  
research carried out by : P.F.C. Krekel

---

**ABSTRACT (ONGERUBRICEERD)**

This report presents the results of the feasibility study investigating the characteristics of complex Zernike moments and their application in translation-, scale- and rotation-invariant object recognition problems. The complex Zernike moments are used as characterising features in a neural network based target recognition approach for the classification of objects in images recorded by sensors mounted on an airborne platform. The complex Zernike moments are a transformation of the image by the projection of the image onto an extended set of orthogonal polynomials.

The emphasis of this study is laid on the evaluation of the performances of Zernike moments in relation with the application of neural networks. Therefore, three types of classifiers are evaluated: a multi-layer perceptron (MLP) neural network, a Bayes statistical classifier and a nearest-neighbour classifier. Experiments are based on a set of binary images simulating military vehicles extracted from the natural background. From these experiments the conclusion can be drawn that complex Zernike moments are efficient and effective object characterising features that are robust under rotation of the object in the image and to a certain extent under varying affine projections of the object onto the image plane.

---

**Codes**

---

**Avail and/or  
Special****List****A-1**

---

rapport no.	:	FEL-92-A394
titel	:	Zernike Momenten en Rotatie Invariante Objektherkenning
		Een Neurale Netwerken Georiënteerde Haalbaarheidsstudie
auteur(s)	:	P.F.C. Krekel
instituut	:	Fysisch en Elektronisch Laboratorium TNO
datum	:	December 1992
hdo-opdr. no.	:	A92KL632
no. in iwp '92	:	708
onderzoek uitgevoerd o.l.v.	:	P.F.C. Krekel
onderzoek uitgevoerd door	:	P.F.C. Krekel

---

#### SAMENVATTING (ONGERUBRICEERD)

Dit rapport beschrijft de resultaten van de haalbaarheidstudie Neurale Netwerken voor een RPV Monitor. Binnen dit onderzoek is gekeken naar de eigenschappen van complexe Zernike momenten en toepassingsmogelijkheden binnen de context van translatie-, schaal- en rotatie-invariante objektherkenning. De complexe Zernike momenten dienen als karakteristieke kenmerken voor het herkennen van objecten in beelden opgenomen door middel van sensoren die geplaatst zijn op een vliegend platform. De eigenlijke Zernike momenten zijn een transformatie van het beeld door middel van een projectie op een uitgebreide set orthogonale polynomen.

De nadruk van het onderzoek ligt op de evaluatie van Zernike momenten in relatie met neurale netwerken als classificatiemechanisme. Met dit doel zijn drie typen klassificatoren uitgetest, te weten een multi-layer perceptron neuraal netwerk, een nearest-neighbour klassificator en een Bayes schatter. Bij het uitvoeren van de experimenten is gebruik gemaakt van een database van beelden met contouren van militaire voertuigen. Uit de experimenten blijkt dat de transformatie van een beeld in complexe Zernike momenten een efficiënte en effectieve wijze is om object-contour kenmerken te karakteriseren zowel onder rotatie van het object in het beeldvlak als in zekere mate bij verandering van de projectierichting van het object in het beeld.

---

**CONTENTS**

<b>ABSTRACT</b>	<b>2</b>
<b>SAMENVATTING</b>	<b>3</b>
<b>CONTENTS</b>	<b>4</b>
<b>LIST OF FIGURES</b>	<b>6</b>
<b>LIST OF IMAGES</b>	<b>7</b>
<b>1. AUTOMATIC TARGET RECOGNITION</b>	<b>8</b>
1.1 Introduction	8
1.2 Automatic Target Recognition Scheme	9
1.3 Invariances	11
1.4 Problem definition	12
<b>2. GEOMETRIC AND ZERNIKE MOMENTS AND THEIR APPLICATIONS</b>	<b>14</b>
2.1 Moments and Functions of Moments	14
2.2 Geometric Moments	15
2.3 Zernike Moments	16
<b>3. IMPLEMENTATION</b>	<b>21</b>
3.1 Zernike Moments and Complexity	21
3.2 Functional Description	23
3.3 Active Vision Library Functions	26
<b>4. EXPERIMENTS</b>	<b>29</b>
4.1 Database Description	29
4.2 Neural Network based Classifiers	32
4.3 Neural Networks and Related Techniques	36
4.4 Conventional Statistical Classifiers	38

---

4.5	Classification Results	40
5.	EXPERIMENTS DESCRIPTION AND SIMULATION RESULTS	42
5.1	Automatic Target Recognition Scheme	42
5.2	Classification Experiments within the RPV Monitor Context	43
5.3	Performance comparisson between different classifiers	45
5.4	Zernike Moments and Neural networks	57
5.5	Generalisation	57
6.	CONCLUSIONS, SUGGESTIONS FOR FURTHER RESEARCH AND CONCLUDING REMARKS	59
6.1	Conclusions	59
6.2	Suggestions for further Research and Concluding Remarks	60
	REFERENCES	62
	IMAGES	67
	APPENDIX A: TRAINING NEURAL NETWORKS	
	APPENDIX B: ACTIVE VISION USER COMMANDS	

## LIST OF FIGURES

Figure 1.1	Schematic overview of the different subsystems of an automatic target recognition system	10
Figure 2.1:	Radial polynomials $R_{nm}$ for various values of the order $n$ with $m$ set to 0	16
Figure 4.1	Relations between the sensor viewing direction, the ground plane onto which an object is positioned and the angles $\alpha$ and $\beta$	31
Figure 4.2	Schematic view of a three layer neural network	39
Figure 5.1.	Absolute values of the first 20 Zernike moment of a test image	44
Figure 5.2.	Zernike moments of the 5 different objects in the database	47
Figure 5.3.	Zernike moments of 6 different object representations	48
Figure 5.4.	Influence of image noise and viewing direction on extracted Zernike moments	49
Figure 5.5	Classification results related to the number of features in the input vector using Zernike Complex moments as invariant features	50
Figure 5.6	Classification results of Multi Layer Perceptron neural networks	52
Figure 5.7	Classification results of Multi Layer Perceptron neural networks related to the signal to noise ratio of the images	54
Figure 5.8	Classification results related to the viewing direction	55

---

LIST OF IMAGES

Image 3.1	Transformation of coordinate system from Cartesian to polar	68
Image 3.2	Zernike complex moments templates	69
Image 3.3	Zernike complex moments templates	70
Image 4.1	Database example of the significance of the test image with respect to a real-life object	71
Image 4.2	Database example of the relative translation, scaling and rotation of the projection of an object onto the image plane	72
Image 4.3	Database example of images of one and the same object with different turret positions	73
Image 5.1	Translation and scaling of an object in the image plane	74
Image 5.2	Database images: topview onto five different objects	75
Image 5.3	Influence of the image noise on the object segmentation	76
Image 5.4	Database example of images of one and the same object viewed from different viewing directions	77
Image 5.5	Database example of images of one and the same object viewed from different viewing directions	78

## 1. AUTOMATIC TARGET RECOGNITION

### 1.1 Introduction

Surveillance and reconnaissance operations are of great importance in any imaginable conflict situation but also in complex and politically sensitive circumstances as arms reduction verification operations and humanitarian operations. Within this context, reconnaissance platforms and satellites are widely used for surveillance tasks on a global scale. On a more local scale, the application of Remotely Piloted Vehicles emerge as an efficient and effective alternative. The flexibility in its use and the cost effectiveness have resulted that in various countries RPV systems have been incorporated into the Recce patrols [Hooton and Munson, '92]. These platforms constantly monitor large areas of the world. A never lasting stream of information, embedded in recorded images, must be extracted, analysed and interpreted by human operators. Therefore, many of today's military platforms would benefit greatly from an automatic object recognition capability in a system that is sophisticated enough that it can substitute the man in the loop, i.e. the human operator who is still necessary or mandatory for the final identification, verification, or last check in today's semi-autonomous recognition systems.

Rapid developments in sensor technology and signal processing algorithms and hardware have made it possible to equip platforms, and especially RPV systems, with advanced sensor systems consisting of combinations of visible light and infrared camera's or FLIR and CO<sub>2</sub>-laser radar. The images or video generated by the sensor systems are broadcasted to a control centre on the ground or returned back by the platform recorded on tape. This information must be processed and analysed by human operators. The large amount of information that becomes available and the possibly stressing conditions in which the analysis and identification work has to be performed, justify any attempt to automate this process.

In this document we present the evaluation results of the performance of Zernike moments, a particular algorithm that may serve as one of many functional blocks in an experimental, yet to be developed automatic target recognition system. As will be explained later, a target recognition system exits of several different stages. The algorithm described in this document is a solution to the problem of translation, scale and rotational variances of the appearance of objects in an image [Dudani et al., '77]. These variances are a natural effect in the images since a sensor platform may



fly on different altitudes or make use of different focal lengths of the camera resulting in varying object dimensions. Furthermore, a platform may approach objects from different angles resulting in varying rotational positions and affine projections of objects onto the image plane.

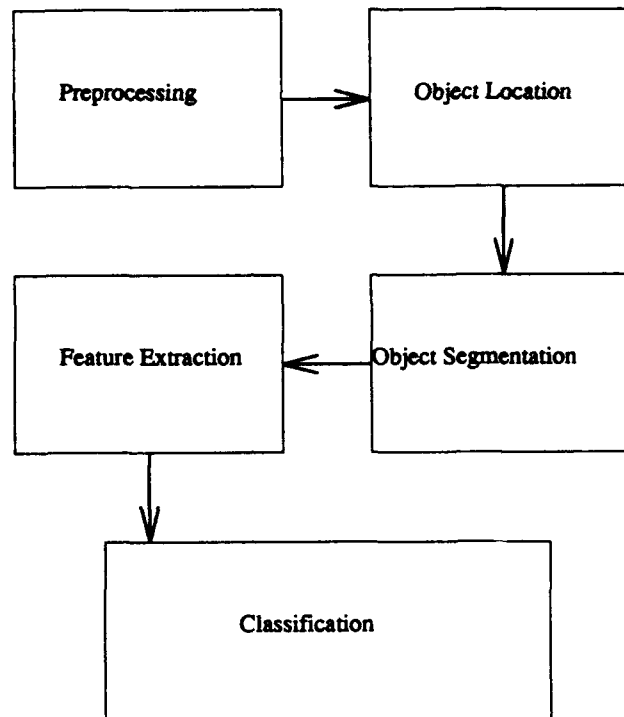
## 1.2 Automatic Target Recognition Scheme

The aim of a vision-based automatic target recognition system is the classification or identification of located objects in an image or sequence of images. In this context, classification can be considered as assigning a class label to a located object. Class labels identify and discriminate different classes according to a given taxonomy. The definition of a class or subclass is based on the selection of a set of features or characteristics that are unique and consistently present in all members of the class.

In the same way, recognition can be regarded as the process of the identification of a unique member (instance) of a given class. Recognition is based on the identification of unique characteristics of one specific object and lies beyond the scope of the project described in this document.

The target recognition process may be subdivided into five stages: preprocessing, object location, object segmentation, feature extraction and object classification. The different stages in the processing scheme and the relations between the different stages is depicted in Figure 1.1.

The aim of the image preprocessing stage is to optimise the starting conditions of the classification trajectory. Knowledge about sensor characteristics or signal propagation, for example, may be used to improve the contrast between the object and the background. One frequently used preprocessing filter in image processing is the median filter and the Gaussian filter. The median filter substitutes each pixel by the median from a small neighbourhood. The Gaussian filter results in a weighted averaging where the weighting coefficients are effectively a 2-dimensional Gaussian. Besides local image operators, there are global operators influencing all image points. An example of a global operator is contrast enhancement by histogram averaging.



**Figure 1.1** Schematic overview of the different subsystems of an automatic target recognition system.

The second stage in the classification trajectory is the location of potentially interesting objects in the image. Texture information or brightness information are frequently used to select interesting areas that have to be further processed. Solving the object location problem is far from trivial. Moving objects may be detected by evaluating the optical flow of pixels in a sequence of images. Pixels representing the object can be distinguished from background pixels because object pixels have different optical flow vectors. An other approach may be based on combining multi-spectral images, e.g. visible light and infrared images. Hot-spots (large concentration of thermal radiation) in infrared images may give clues for the position of objects in the visible light images.

After locating the object position in the image, the object has to be separated from its background as accurate as possible. Again, brightness information can be used in combination with contour information. Most artificial or man-made objects have convex borders that can be separated from

concave borders. Edges may be found from a grey scale image by filtering the image with an edge detector and thresholding the result. The accuracy of this process influences to a large extent the reliability of the further processing.

The next step in the classification trajectory is the calculation of geometric and spectral features of each segmented region. Geometric features include the average intensity value of the segment, the occupied area and the ratio of its length to its width. This information can be captured in the segment moments. Also the curvedness of the segment border can be considered. This information can be extracted from so-called Fourier descriptors. Spectral information can give a measure of the texture of the region. In general, we can say that we are looking for those object features that are invariant for translation, scaling and rotation of the object in the image since those parameters are very difficult to control in our application. For an extended survey about invariant features or representations of image objects, see for example [Toet, '92].

Finally, the object classification is based on the statistical evaluation of the features attributes values and probabilities. Nearest-neighbour and minimum-mean-distance classifiers or neural networks can be used to subdivide the multi-dimensional feature space in subspaces by decision boundaries [Duda and Hart, '73][Schalkoff, '92]. Each subspace or set of subspaces represents a class. If a feature vector is lying in one of the subspaces, the related class label is assigned to the object. On the other hand, if different sources of evidence for a particular object are available, e.g. in a multiple-sensor system, other techniques like the Dempster-Schafer theory [Shafer, '76] may be used in combination with a knowledge-based system to perform the classification task.

### 1.3 Invariances

As stated above, the problem of invariances is our main point of interest. In literature, two alternative approaches towards the problem of invariances can be found that have shown to be reasonably successful: the application of moments making use of moment invariants [Hu, '62] and boundary encoding making use of Fourier boundary descriptors [Persoon and Fu, '77]. The latter is also known to be invariant under affine transformations [Arbter, '89][Arbter, Snyder, Burkhardt and Hirzinger, '90]. Both techniques suffer from the disadvantage of being computationally intensive. Recently, a fast alternative has been developed and published by Schau [Schau, '92]. It is based on analysing basic 2 by 2 elements of a binary image. The number of basic elements is used to define a distance measure between the unknown object and a set of

known prototypes. This technique is interesting because of its simplicity and its potential for fast implementation.

In spite of its complexity, we have chosen to investigate the invariance properties of the complex Zernike moments because of their promising results presented in the literature. To obtain the Zernike moments we need a radial symmetric transformation which may be accomplished by a combination of a circular Fourier transform and a radial Mellin transform [Grace and Spann, '91][Sheng and Arsenault, '86][Sheng and Duvernoy, '86]. In our experiments, however, we made use of predefined templates which reduced the computations to simple image multiplication. The complexity of the algorithm can be further reduced by performing a polar transform on the image followed by simple vector operations to obtain the desired moments. This will be discussed in one of the following sections.

#### 1.4 Problem Definition

The goal of the project *Haalbaarheidstudie Neurale Netwerken* is to demonstrate that artificial neural networks are a powerful paradigm and provide useful building blocks to construct a robust and reliable target recognition system.

The target recognition processing scheme, as presented above, exists of several modules. In theory, neural networks can be used to implement operations for each individual module [Roth, '90]. Though all modules are more or less equally important in an actual system, we have decided to focus on the feature extraction and classification modules only. Other topics are covered by various TNO-FEL research projects.

The location problem for example, can be solved by integrating multi-sensor information originating from visible light and infrared sensors. An infrared image can be used for object location by detection of hotspots, characteristic radiation patterns in the infrared part of the spectrum. Moving objects may be located by searching for irregularities in the optical flow field calculated from a sequence of images. Moving objects induce optical flow vectors that are different from the vectors due to the motion of the sensor platform [Beck, '92].

In our approach, neural network algorithms are embedded in the classification module only. Neural networks may be used for feature extraction also [Perantonis and Lisboa, '92], but since we want to have complete control over the features that are used by the system, the feature extraction concepts within this most important module, will be based on classical techniques.

This document starts with the introduction of the geometric and Zernike moments in Section 2. The geometric moments will be applied to resize the object in the image being processed to a uniform area and to place the resulting object representation in the centre of the image. The Zernike moments will be evaluated to capture rotation invariant object features in a consistent manner. These features will be used to classify the object, regardless of its rotated position in the scene.

The evaluation of Zernike moments is a computationally complex task. In Section 3, we describe how the geometric and Zernike moment extraction process can be implemented efficiently in software. Furthermore, an alternative solution is presented for real-time implementations. In Section 4 the context and the setup of the experiments to evaluate the Zernike moments as invariant features for object recognition are presented. An overview of the database comprising test images of military vehicles is given and the different classification approaches, both based on neural networks and classical techniques, are given. Finally, in Section 5, the results of the numerous experiments are presented and conclusions are drawn related to the usefulness of the application of Zernike moments within the context of automatic target recognition.

The research, of which the results are presented in this report, has been done within the framework of the feasibility study RPVM-NN which has been carried out for the HOE-AI project *Haalbaarheidstudie Neurale Netwerken* (Gaining Experience with Artificial Intelligence) under responsibility of the Development Centre for the Automation of Weapon and Command systems of the Directorate of Economy and Finance of the Royal Netherlands Army (DEBKL/DCAWACO).

## 2. GEOMETRIC AND ZERNIKE MOMENTS AND THEIR APPLICATIONS

In this section some fundamentals related to moments and functions of moments are presented. These fundamentals will be used in the following sections to explain the design of the translation, scale and rotation invariant target recognition system evaluated in this report.

In Section 2.1 an overview of recent literature related to various types of moments is presented. In Section 2.2 the geometric moments are introduced where in Section 2.3 the emphasis lies on the complex Zernike Moments.

### 2.1 Moments and Functions of Moments

Moments and functions of moments are powerful object characterising features in classification problems, involving invariant recognition of 2-dimensional patterns in an image. Moments are actually projections of the image parameter function onto a set of polynomials where the parameters are the pixel coordinates. If we use a set of mutually orthogonal polynomials, we obtain a set of uncorrelated, non-redundant characteristics. Fundamentals related to the theory of moments can be found in [Gonzalez and Wintz, '77] and [Rosenfeld and Kak, '82].

The concept of moment invariants for pattern recognition was introduced by Hu in 1962 [Hu, '62]. Hu has derived a set of invariant moments that has the property of being invariant under image translation, scaling and rotation. An example of the application of moments in a problem context similar as ours is given in [Dudani et al., '77]. Teague [Teague, '80] suggested the use of orthogonal moments based on the theory of orthogonal polynomials and introduced the Zernike moments which provide independent moment invariants to an arbitrarily high order. Other moments are the pseudo-Zernike moments, rotational moments [Boyce and Hossack, '83], complex moments [Abu-Mostafa and Psaltis, '84][Abu-Mostafa and Psaltis, '85] and the Legendre moments based on the Legendre polynomials [Teh and Chin, '88].

Satisfactory experimental outcomes resulted in considerable attention in literature up until now. For example, a revised fundamental theorem concerning moments is given in [Reiss, '91] and an extended survey of moment based techniques for recognition is presented in [Prokop and Reeves, '92].

In the context of pattern recognition, moment invariances can be considered as reliable and robust features if their values are insensitive to image noise. Detailed results concerning noise sensitivity and an approach to select an optimal set of moment features are given in [Teh and Chin, '88] and

[Khotanzad and Hong, '90], respectively. In theory, most of the image information can be reconstructed by using a sufficiently large number of a particular set of image moments. This fact can be used to select the appropriate number of object features necessary to solve a classification problem.

Having captured the image information into a set of moments, still leaves us with the actual classification problem. In [Khotanzad and Lu, '90] moments are used in combination with a multi-layer perceptron neural network as a promising solution to the translation, scaling and rotation invariant classification of objects in an image. Grosso modo, we have adapted their approach in building an invariant object recognition system. In this section the geometric and Zernike moments will be introduced where in Section 4 the multi-layer perceptron neural network will be presented.

## 2.2 Geometric Moments

The geometric or regular moments are projections of the image function onto the monomial  $x^p y^q$  where  $x, y$  are the image coordinates. The regular moments  $M_{pq}$  of order  $(p+q)$  of the image function  $f(x, y)$  are defined as

$$M_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy, \quad (2.1)$$

where  $p, q = 0, 1, 2, \dots, \infty$  and  $M_{pq}$  is the  $(p+q)$ th order moment of the continuous image function  $f(x, y)$ . Assuming that the image function  $f(x, y)$  is piecewise continuous and has a compact support, moments of all order exist and the infinite set of moments uniquely determines  $f(x, y)$ . Reversely, the moments are themselves uniquely determined by  $f(x, y)$ . For digital images the integrals in Equation 2.1 are replaced by summations and  $M_{pq}$  becomes

$$M_{pq} = \sum_x \sum_y x^p y^q f(x, y). \quad (2.2)$$

As mentioned above, the definition of regular moments has the form of the projection of the image function  $f(x, y)$  onto the monomial  $x^p y^q$ . The disadvantage of using geometric moments is that the basis set  $\{x^p y^q\}$ , while complete, is not orthogonal. Therefore, information captured in these moments is redundant.

Regular moments characterise the spatial distribution of an image. Image statistics such as the centre of mass, and moments of inertia in an image can be directly calculated from these moments. The image centre of mass or centroid location  $(\bar{x}, \bar{y})$  can be obtained from the first and zeroth moment and is given by

$$\bar{x} = \frac{M_{10}}{M_{00}}, \quad \bar{y} = \frac{M_{01}}{M_{00}}. \quad (2.3)$$

The zeroth and first order regular moments are important for our object classification system since they will later be used in a preprocessing stage to centre the object in the image. In this way translation invariance is achieved.

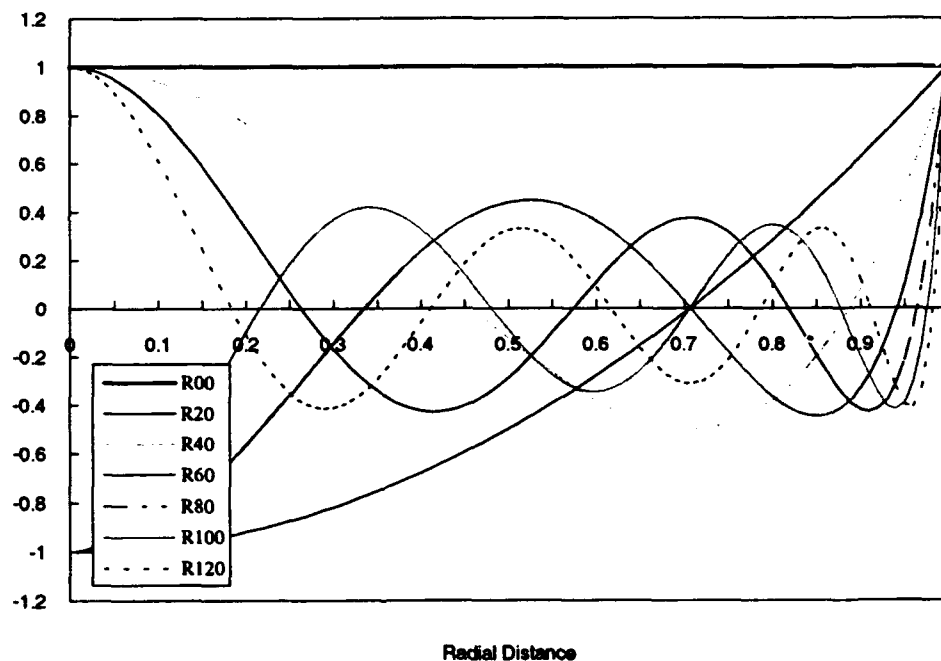


Figure 2.1: Radial polynomials  $R_{nm}$  for various values of the order  $n$  with  $m$  set to 0. Note that  $R_{nm}(1)=1$  for all  $n$  and  $m$ .

### 2.3 Zernike Moments

The disadvantage of the regular moments as presented in Section 2.1 is that the basis set  $x^p y^q$  is not orthogonal. Therefore, features defined on functions of the basis set are not optimal with respect to information redundancy and other characteristics. Uncorrelated features can be derived



by making use of orthogonal functions. To this end, the Zernike set of complex polynomials  $\{V_{nm}(\rho, \theta)\}$  is introduced [Zernike, '34]. This set constitutes a complete orthogonal set defined over the interior of the unit circle, i.e.  $x^2 + y^2 \leq 1$ .

Zernike moments  $A_{nm}$  are the projection of the image function  $f(x, y)$  onto this set of orthogonal basis functions  $\{V_{nm}(\rho, \theta)\}$ . The orthogonal basis functions can be written as

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \exp(jm\theta). \quad (2.4)$$

Here, the orthogonal basis function is written as the product of a radial polynomial  $R_{nm}(\rho)$  and a harmonic function of the angular coordinate (phase component). The definition of the radial polynomial  $R_{nm}(\rho)$  is rather complex and is derived in [Born and Wolf, '75]:

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|)/2} \frac{(-1)^s [(n-s)!] \rho^{n-2s}}{s! \left(\frac{n+|m|}{2} - s\right)! \left(\frac{n-|m|}{2} - s\right)!}. \quad (2.5)$$

The normalisation of the polynomial has been chosen so that for all permissible values of  $n$  (degree) and  $m$  (angular dependence or repetition),

$$R_{nm}(1) = 1. \quad (2.6)$$

Several examples of radial polynomials of various combinations of  $n$  and  $m$ , normalised to  $R_{nm}(1)=1$ , are given in Figure 2.1.

In Table 2.1 the explicit form of the polynomials for the first few values of the indices  $n$  and  $m$  is given. The polynomials  $V_{nm}(x, y)$  are defined for order  $n$  with repetition  $m$  where  $n = 0, 1, 2, \dots, \infty$  and  $m$  takes on positive and negative integer values subject to the conditions  $n - |m| = \text{even}$  and  $|m| \leq n$ . The Zernike moments of order  $n$  with repetition  $m$  for a continuous image function,  $f(x, y)$ , that vanishes outside the unit circle, is

$$A_{nm} = \frac{n+1}{\pi} \iint_{x^2+y^2 \leq 1} f(x, y) V_{nm}^*(\rho, \theta) dx dy. \quad (2.6)$$

Here, the symbol \* denotes the complex conjugate. For a digital image, the integrals in Equation 2.6 are replaced by summations resulting in

$$A_{nm} = \frac{n+1}{\pi} \sum_x \sum_y f(x,y) V_{nm}^*(\rho, \theta), \quad x^2 + y^2 \leq 1 \quad (2.7)$$

To compute the Zernike moments of a given image, the centre of the image is taken as the origin and pixel coordinates are mapped to the range of the unit circle. Those pixels falling outside the unit circle are omitted during the computation.

m \ n	0	1	2	3	4	5	6
0	1		$2\rho^2 - 1$		$6\rho^4 - 6\rho^2 + 1$		$20\rho^6 - 30\rho^4 + 12\rho^2 - 1$
1		$\rho$		$3\rho^3 - 2\rho$		$10\rho^5 - 12\rho^3 + 3\rho$	
2			$\rho^2$		$4\rho^4 - 3\rho^2$		$15\rho^6 - 20\rho^4 + 6\rho^2$
3				$\rho^3$		$5\rho^5 - 4\rho^3$	
4					$\rho^4$		$6\rho^6 - 5\rho^4$
5						$\rho^5$	
6							$\rho^6$

Table 2.1: The radial polynomials  $R_{nm}(\rho)$  for  $m \leq 6, n \leq 6$ .

The magnitudes of the Zernike moments are invariant under image rotation. Given a rotation of an image through an angle  $\phi$ , the relationship between  $A'_{nm}$  and  $A_{nm}$ , the Zernike moment of the rotated image and the unrotated one, is given by

$$A'_{nm} = A_{nm} \exp(-jm\phi) \quad (2.8)$$

This relation shows that Zernike moments have simple rotational transformation properties; each Zernike moment merely acquires a phase shift on rotation. From this relation the desired property follows: The magnitude  $|A_{nm}|$  of the Zernike moments of a rotated image function remain identical to those before rotation. Hence, the magnitude  $|A_{nm}|$  of the Zernike moment can be taken as a rotation invariant feature of the underlying image function.

Order	Moments	No. of Moments
0	$A_{00}$	1
1	$A_{11}$	1
2	$A_{20}, A_{22}$	2
3	$A_{31}, A_{33}$	2
4	$A_{40}, A_{42}, A_{44}$	3
5	$A_{51}, A_{53}, A_{55}$	3
6	$A_{60}, A_{62}, A_{64}, A_{66}$	4
7	$A_{17}, A_{73}, A_{75}, A_{77}$	4
8	$A_{80}, A_{82}, A_{84}, A_{86}, A_{88}$	5
9	$A_{91}, A_{93}, A_{95}, A_{97}, A_{99}$	5
10	$A_{10,0}, A_{10,2}, A_{10,4}, A_{10,6}, A_{10,8},$ $A_{10,10}$	6
11	$A_{11,1}, A_{11,3}, A_{11,5}, A_{11,7}, A_{11,9},$ $A_{11,11}$	6
12	$A_{12,0}, A_{12,2}, A_{12,4}, A_{12,6}, A_{12,8},$ $A_{12,10}, A_{12,12}$	7

Table 2.2: List of Zernike moments and their corresponding number of features from order 0 up to order 12.

The image function  $f(x,y)$  can be expanded in terms of the Zernike polynomials over the unit disk as

$$f(x,y) = \sum_{n=0}^{\infty} \sum_{m=-\infty}^{\infty} A_{nm} V_{nm}(x,y), \quad n-|m| = \text{even}, \quad |m| \leq n, \quad (2.9)$$

where the Zernike moments  $\{A_{nm}\}$  are computed as in Equation 2.7.

If the series expansion is truncated at a finite order  $N$ , then the truncated expansion is the optimum approximation  $\hat{f}(x,y)$  to  $f(x,y)$ :

$$\hat{f}(x,y) = \sum_{n=0}^N \sum_m A_{nm} V_{nm}(x,y), \quad n-|m| = \text{even}, \quad |m| \leq n. \quad (2.10)$$

Though image reconstruction from Zernike moments is not the goal of our algorithm, Equation 2.10 can be used to determine the optimal order of moments that capture the most important characteristic features of the object to be classified.

A simple error function defined as the difference between the original image function  $f(x,y)$  and the reconstructed image function  $\hat{f}(x,y)$  can be used to determine the optimal order  $N$  of the moments. From [Teh and Chin, '88] it is known that under poor signal to noise conditions, the reconstructed image degenerates when the maximal order  $N$  exceeds a certain boundary. In practice, it is not possible to determine an optimal subset of complex Zernike moments for classification purposes since each image has its unique optimal decomposition into orthogonal components.

In case of selecting the order of the Zernike moments to be equal to 12, the total number of moments is equal to 49 (See Table 2.2). In this case, an object is characterised by 49 features,  $A_{0,0} \dots A_{12,12}$ . It are those numbers, grouped into a feature vector, that will be used by a target recognition system as input for the classification process.

### 3. IMPLEMENTATION

To be able to perform the experiments to investigate the performance of Zernike moments as invariant features for object recognition we have implemented the equations given in Section 2 in a straightforward way. All code is written in the programming language C. For some basic image I/O routines and image preprocessing functions we have made use of the image processing library Sunvision IP [SunVision, '91]. Clusters of functions are combined in several programs resulting in a set of command line user commands. These commands can be used to perform relevant operations on separate images in command line mode. The individual functions itself may be used in newly to develop programs in the same way as the functions available in the SunVision library. First, in Section 3.1 some basic considerations concerning implementation aspects, especially speed, are given. Next, in Section 3.2 the command line functions are described. Finally, in Section 3.3 the underlying basic functions are presented. For a more detailed description of the functions we refer to Appendix B.

#### 3.1 Zernike Moments and Complexity

An image can be considered as an discrete array of values of an uniform type. A complex Zernike moment of an given order and repetition of this image can be obtained by evaluating Equation 2.7 in Section 2. From this equation it follows that we have to evaluate for each pixel  $x,y$  an orthogonal basis function  $V_{nm}(\rho,\theta)$  where  $n$  is the order and  $m$  the repetition of the Zernike moment of consideration. Hence, each position index  $x,y$  has to be transformed to a range value  $\rho$  and an angle  $\theta$ .

Since, in general, we are interested in a extended set of Zernike moments, several orthogonal basis functions are evaluated for each individual array index  $x,y$ . Therefore, for each array index, the transformation from Cartesian  $x,y$  coordinates to polar  $\rho,\theta$  coordinates is done in advance: Two arrays are generated, a *range* image and a *phase* image, of the same dimension as the image to be processed. In the range image, at index  $x,y$ , the range from the image centre to pixel  $x,y$  is stored and in the phase image, at index  $x,y$ , the phase rotation from an imaginary coordinate system positioned at the image centre is stored.

Next, for each Zernike order  $n$  and repetition  $m$ , and each polar coordinate  $\rho$  and  $\theta$ , the orthogonal basis function  $V_{nm}(\rho,\theta)$  has to be evaluated. As can be seen from Equation 2.4,  $V_{nm}(\rho,\theta)$  is a product of a radial polynomial and a phase component. The complexity of the radial component depends on the order of the Zernike moment of consideration. The higher the order,

the larger the number of terms in the polynomial that have to be evaluated (see Table 2.1). Therefore, the processing time, required to determine the complex Zernike moment of an particular order  $n$ , increases for higher order moments. A first approach to increase processing speed is based on the calculation of Zernike templates. As can be seen from Equation 2.7, the determination of a Zernike moment  $A_{nm}$  can be considered as a dot product between image array  $f$  and basis function array  $V_{nm}$ . This basis function array will be denoted as a Zernike moment *template* of order  $n$  and repetition  $m$ . The templates can be computed for each order and repetition in advance. A template is computed by evaluating for each array index  $x,y$  the orthogonal basis function  $V_{nm}(\rho,\theta)$  of Equation 2.4 making use of the predetermined range and phase arrays to transform Cartesian coordinates into polar coordinates (table lookup). In Image 3.2 and 3.3 templates for various Zernike moments are displayed. The influences of the harmonics controlled by the repetition coefficients can clearly be seen.

Note: Since we only use absolute values of the complex Zernike moments, processing time can also be reduced by determining only the absolute part. However, in our implementation, the complex template array is subdivided into an array containing the real part, an array containing the imaginary part and a third array comprising the absolute value of the complex template.

The advantage of the template based approach lies in the fact that the computation time of each individual Zernike moment is independent of the order and repetition: The number of operations required to process the images is fixed. However, the time needed to calculate the templates in advance does depend on the specific order.

The most cost efficient solution one can think of to obtain Zernike moments of an given image, is based on reducing the multiplication of two images to the multiplication of two vectors. Since for recognition purposes we are not interested in the phase information of the Zernike moments, we have already omitted this information in the approach described above (we do only need phase information for reconstruction purposes). By doing this, we actually multiply each image pixel positioned on the same concentric circle of radius  $\rho$ , by a radial polynomial of order  $n$  and repetition  $m$  at radial distance  $\rho$ . When we intergrate the pixels lying on one and the same concentric circle, before multiplying with the radial polynomial, we do reduce the number of multiplications dramatically. One way of summing pixels lying on a concentric circle is first transforming the image coordinates from Cartesian to polar coordinates. In this representation, pixels lying in the same row all have the same relative angle with respect to a imaginary

Cartesian coordinate system positioned at the image centre, where pixels lying in the same column all are lying at the same radial distance from this coordinate system centre.

After transforming the image coordinates from Cartesian to polar, the integration of the pixels lying on the same concentric circle can be accomplished by simply summing the pixels lying in one and the same column. An example of transforming an image from Cartesian coordinates to polar coordinates is given in Image 3.1.

There is a lot of computational overhead in transforming the image's coordinate system from Cartesian to polar. Especially when we make use of subsampling interpolation techniques. However, the reduction in computational complexity switching from matrix-matrix multiplication to vector-vector multiplication is of such order that the overall complexity reduction may result in an much faster overall implementation of the Zernike moments generation function.

Note: For accuracy reasons, it may be necessary to increase the image dimensions transforming from Cartesian to polar coordinates resulting in a vector product of a larger dimension than the image dimension.

### 3.2 Functional Description

In this subsection we present a set of command line functions related to the extraction of Zernike complex moments from an image array. For some of the functions described, a standard implementation and a fast implementation based on templates are given. During the experiments we have made use of the fast implementation. The numerical results of both implementations do differ slightly due to round-off errors in the Zernike template images. However, this has no effect on the classification process.

#### 3.2.1 `zernike_moment`

The Zernike moments of an image can be calculated by the user command `zernike_moment`. The command generates the Zernike moments of the input image and stores the results in an output file. For each moment, the real, imaginary and absolute value of the complex Zernike moment are successively stored. The moments are calculated from a specified minimal order value up to a specified maximal order value for all valid repetition values. The range of the valid moment order parameters lies between 0 and 20. For a complete synopsis of the command `zernike_moment`, see Appendix B.

The command *zernike\_moment* is based on the functions *av\_radial\_dist\_image*, *av\_angular\_image* and the function *av\_zernike\_moment*. These functions will be described in the following section.

### 3.2.2 *zernike\_fmoment*

The implementation of the command *zernike\_moment* is relatively slow. It takes about 1 second on average for each calculated moment on a Sun Sparc station 2. Determining all moments up to order 20 (121 coefficients), this processing takes about 2 minutes! A faster implementation of the command *zernike\_moment* is available via the command *zernike\_fmoment*. The command generates the Zernike moments of the input image and stores the results in an output file. For each moment, the real, imaginary and absolute value of the complex Zernike moment are successively stored. The calculation is based on the predefined Zernike moment templates for much faster calculation. The moments are calculated from a specified minimal order value up to a specified maximal order value for all valid repetition values. The range of the valid moment order parameters lies between 0 and 20. For a complete synopsis of the command *zernike\_fmoment*, see Appendix B.

The command *zernike\_fmoment* is based on the function *av\_new\_zernike\_moment*. This function will be described in the following section.

### 3.2.3 *zernike\_reconstruct*

An image can be (partly) reconstructed from its moments. The command *zernike\_reconstruct* generates an image based on the complex Zernike moments. The quality of the reconstruction depends on the number of moments available. The image is reconstructed making use of a set of moments ranging from a specified minimal order up to a specified maximal order. The range of valid moment order parameters lies between 0 and 20. The resulting output image is the sum of an input image, possibly empty, and the generated image based on the moment coefficients. For a complete synopsis of the command *zernike\_reconstruct*, see Appendix B.

The command *zernike\_reconstruct* is based on the functions *av\_radial\_dist\_image*, *av\_angular\_image* and the function *av\_reconstruct*. These functions will be described in the following section.



### 3.2.4        *zernike\_freconstruct*

As is the case for the command *zernike\_moment*, a faster version of the command *zernike\_reconstruct* is available via the command *zernike\_freconstruct*. The command *zernike\_freconstruct* generates an image based on the complex Zernike moments. The quality of the reconstruction depends on the number of moments available. The image is reconstructed making use of a set of moments ranging from a specified minimal order up to a specified maximal order. The range of valid moment order parameters lies between 0 and 20. The resulting output image is the sum of an input image, possibly empty, and the generated image based on the moment coefficients. The calculation is based on predefined Zernike moment templates for fast calculation. For a complete synopsis of the command *zernike\_freconstruct*, see Appendix B.

The command *zernike\_freconstruct* is based on the function *av\_new\_reconstruct*. This function will be described in the following section.

### 3.2.5        *zernike\_template*

As mentioned above, the commands *zernike\_fmoment* and *zernike\_freconstruct* make use of predefined templates. These templates can be generated by the command *zernike\_template*. The command *zernike\_template* generates template images in the .VFF file format [SunVision, '91] of Zernike polynomials making use of a set of moments ranging from a specified minimal order up to a specified maximal order. The range of valid moment order parameters lies between 0 and 20. For each valid order-repetition combination a separate template is generated and stored in a file. The template consists of three bands. One band to calculate the real part of the Zernike moment, one band to calculate the imaginary part of the Zernike moment and one band to calculate the absolute value of the Zernike moment. This latter band may be omitted since the absolute value can be obtained from the real and imaginary value. For a complete synopsis of the command *zernike\_template*, see Appendix B.

The command *zernike\_template* is based on the functions *av\_radial\_dist\_image*, *av\_angular\_image* and the function *av\_make\_polynomial\_image*. These functions will be described in the following section.

### 3.2.6 `norm_image`

Before the Zernike moments of an image can be determined, the image object has to be centred and scaled to a uniform size. This preprocessing may be performed by the command `norm_image`. The command `norm_image` places the image object into the centre of the image by determining the centre of gravity of the object. Then the image centre is translated to this position. Furthermore, the command normalises the binary (logical) image with respect to the number of non-zero pixels in the image. The desired number of non-zero pixels in the output image can be specified by the user.

The input image must be either an `avBYTE` image with logical values 0 and 255, or an `avFLOAT` image with logical values 0.0 and 1.0. For a complete synopsis of the command `zernike_template`, see Appendix B.

The command `norm_image` is based on the functions `av_smoments` and `av_norm_image`. These functions will be described in the following section.

## 3.3 Active Vision Library Functions

All command line user commands are based on Sunvision library functions and a selection of the Active Vision Library functions described below. The Active Vision library is a set of image processing functions developed by TNO-FEL. A synopsis of each function is given in Appendix B.

### 3.3.1 `av_radial_dist_image`

The function `av_radial_dist_image` generates a radial distance image having the same dimensions as the input image. The pixel values in the radial distance image represent the polar coordinate  $\rho$  measured relative to the coordinate system's centre. Therefore, the rectangular and polar coordinate system centres are translated to the image centre. The generated radial distance image may be used as a look-up table to transform a rectangular or Cartesian coordinate system into a polar coordinate system. The radial distance  $\rho$  of the transformed grid position  $(x,y) \rightarrow (\rho,\theta)$ , is given by `image(x,y)`. The radial distance coordinate  $\rho$  is ranging from 0.0 to 1.0. Pixels lying outside the implicitly defined unit circle are set to -1.0.

### 3.3.2 `av_angular_image`

The function *av\_angular\_image* generates an angular image having the same dimensions as the input image. The pixel values in the angular image represent the polar coordinate  $\theta$  measured relative to the x-axis in a counter clock wise direction. Therefore, the rectangular and polar coordinate system centres are placed into the image centre. The generated angular image may be used as a look-up table to transform a rectangular or Cartesian coordinate system into a polar coordinate system. The angle theta of the transformed grid position  $(x,y) \rightarrow (p,\theta)$ , is given by  $\text{image}(x,y)$ . The angular coordinate theta is ranging from 0.0 to  $2\pi$ .

### 3.3.3 `av_make_polynomial_image`

The function *av\_make\_polynomial\_image* generates a Zernike template image for a complex Zernike polynomial of a predefined order and repetition. The template combines the radial polynomial information with the phase information. The template exists of a real, imaginary and absolute part, all stored in different bands. The real template is stored in band 0, the imaginary template is stored in band 1, and the absolute template is stored in band 2. The template may be used to determine the complex Zernike moment for the given order and repetition of an image having the same dimensions as the template. The real, imaginary and absolute moment value can be obtained by multiplying the image with the appropriate template band and sum the result.

### 3.3.4 `av_zernike_moment`

The function *av\_zernike\_moment* determines the complex Zernike moment of a given order and repetition of the input image. The function recalculates for each pixel value the radial polynomial value and the phase information according to Equation 2.6 in Section 2. This is done by a call to the function *av\_radial\_polynomial* and a sine and cosine transform. The moment calculation is done in the polar coordinate system. The transformation of the rectangular coordinate system into the polar coordinate system is based on a table look-up strategy. Therefore, a radial distance image and an angular image of the appropriate dimensions serve as input arguments to the function.

### 3.3.5 `av_new_zernike_moment`

The function *av\_new\_zernike\_moment* determines, like the previous function, the complex Zernike moment of a given order and repetition of the input image. Therefore, the function make

use of the Zernike template of the correct order, repetition and dimensions. The function multiplies the input image with the appropriate bands (real and complex to determine the real and complex part of the moment variable) of the template and adds the pixels of the resulting image together. This approach is much faster than the approach described for the function *av\_zernike\_moment*. Furthermore, the calculation time of moments of any order are identical where the calculation time of higher orders grow exponentially with the previous approach.

### 3.3.6 *av\_reconstruct*

The function *av\_reconstruct* generates an image based on one single complex Zernike moment coefficient of a predefined order and repetition. For each rectangular grid position the related polar coordinates  $\rho$  and  $\theta$  are determined by looking up these values in the look-up tables angular image and radial distance image. Therefore, a radial distance image and a angular image of the appropriate dimensions serve as input arguments to the function. Given the radial distance value  $\rho$ , the order and the repetition, the radial polynomial value is obtained by a function call to the function *av\_radial\_polynomial*. The reconstruction is based on Equation 2.10 in Section 2. From this equation it follows that the reconstructed pixel value is a weighted sum of sines and cosines of the radial polynomial value.

### 3.3.7 *av\_new\_reconstruct*

The function *av\_new\_reconstruct* generates an image based on one single complex Zernike moment coefficient of a predefined order and repetition. The reconstruction is based on Equation 2.10 in Section 2. To improve the speed of the reconstruction, the approach is based on a predefined template of the appropriate order, repetition and dimensions. The computations are reduced to multiplying the real template image with the real Zernike moment coefficient, multiplying the imaginary template image with the imaginary Zernike moment coefficient and adding the two resulting images together.

### 3.3.8 *av\_radial\_polynomial*

The function *av\_radial\_polynomial* determines the Zernike radial polynomial of a given order and repetition for a specific radial distance. The function is based on Equation 2.5 in Section 2. When the order increases, the evaluation of the function takes longer since more terms of the power series have to be determined. The increase in time is exponentially.

#### 4. EXPERIMENTS

In the previous section we have described, at a functional level, all the programs that had to be developed before we could start any experiments related to the evaluation of Zernike moments and their application as invariant features for object recognition. In this section we focus on all the other issues that are important to be able to generate experimental data and to evaluate the results.

First of all, we need test data, i.e. a series of test images recorded under well defined conditions. Since we had decided from the beginning, only to focus on the feature extraction and object identification module of the target recognition processing scheme, we assume that the problems related to the preprocessing-, object location- and object segmentation-stages, as described in Section 1, are solved. Especially concerning the object segmentation stage, this is an ideal and fairly hypothetical situation.

Second, the actual classification is done by a multi-layer perceptron neural network. In Section 4.2 some fundamentals related to this network are repeated. To evaluate the performance of the neural network as a classifier, we have to compare the network with traditional statistical classifiers like the nearest neighbour classifier and the multidimensional probability density function estimator presented by Parzen [Parzen, '62]. The two classical classifiers we have used for comparison are described in Section 4.3.

##### 4.1 Database Description

To perform experiments under controlled conditions a database of binary images is generated. The database consists of images of 256x256 pixels stored in Sunvision's .VFF file format [SunVision, '91]. Each image represents an object separated from the background. An image pixel is represented by a float value. Since the images are bi-valued, there are only two possible pixel values. The object pixels are indicated by the value 1.0, where the background pixels are indicated by the value 0.0. Each image file comprises exactly one single banded (mono-colour) image. An example of an image from the database and the significances of this image related to an image from the actual object can be seen in Image 4.1.

The objects represented in the images are army vehicles and tanks. The database comprises images of a Leopard 2 tank, a M109 Howitzer, a T-80 main battle tank, a Gepard and a M113 armoured personnel carrier. A list of the database is given in Table 4.1.

The images originate from views taken by a cod camera of models of the objects at a scale of 1:37. The database consists of five different object types. For each type, images are taken from three different viewing positions, resulting in three different viewing directions. A viewing direction can be indicated by two angles representing the relation between the optical axis of the camera and the plane the object is positioned on. The optical camera axis is directed towards the origin of a coordinate system whose x and y axis are lying on the ground plane and the z axis is perpendicular to the ground plane. The angle of the optical axis with the object ground plane is given by  $\alpha$ , where the angle of the projected optical axis onto the groundplane with the orthogonal coordinate system is given by  $\theta$ . Variations in  $\alpha$  from  $90^\circ$  towards  $0^\circ$  effect the viewing direction from *top view* towards *horizontal view*. Variations in  $\theta$  from  $90^\circ$  towards  $-90^\circ$  effect the viewing direction from *front view*, via *side view* towards *back view*. In Figure 4.1 the relations between the viewing direction, ground plane and the angles  $\alpha$  and  $\theta$  are depicted.

As stated above, the database comprises three viewing directions. The first viewing direction is represented by  $\{\alpha=90^\circ, \theta=\emptyset\}$ , which is equal to a top view. For a top view, the angle  $\theta$  is irrelevant. The object may have any orientation in the ground plane, and hence in the image plane. The second viewing direction is represented by  $\{\alpha=60^\circ, \theta=\pm 30^\circ\}$ , which is equal to a top-semi-horizontal view. The value  $\pm 30^\circ$  indicates that the angle  $\theta$  is not known exactly and may vary between  $-30^\circ$  and  $+30^\circ$ . A value for  $\theta$  of  $+30^\circ$  represents a semi front-side view, where an value for  $\theta$  of  $-30^\circ$  represents a semi back-side view. The third viewing direction is represented by  $\{\alpha=30^\circ, \theta=\pm 30^\circ\}$ , which is equal to a semi-horizontal view. Again the angle  $\theta$  may vary between  $-30^\circ$  and  $+30^\circ$ . Decreasing the viewing direction  $\alpha$ , the viewing direction  $\theta$  becomes more and more relevant since the front view of a vehicle in general will differ largely from the side view or back view. Therefore, the range of  $\theta$  is restricted between  $-30^\circ$  and  $+30^\circ$ .

For each viewing direction six different images of the same object are taken. Rotations and small translations (relative to the object size) of the object and different focal distances of the lenses are taken into account. These variations are useful to test the scale, translation and rotation invariant properties of the various algorithms. As an example, six images of one of the objects in the database, taken from a top-view camera position, are depicted in Image 4.2. The relative translation, scaling and rotation of the object in the image plane can clearly be seen. In a real scenario, the different projections of the objects on the image plane may be due to variations in the altitude of the platform the sensors are mounted on, different focal lengths of the cameras,

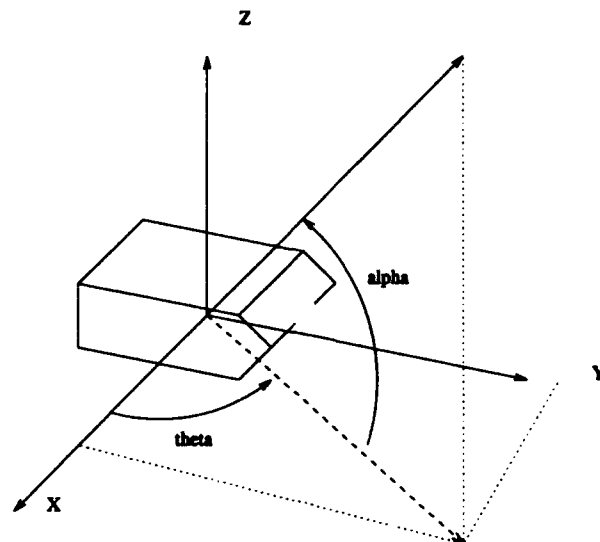


Figure 4.1 Relations between the sensor viewing direction, the ground plane onto which an object is positioned and the angles  $\alpha$  and  $\theta$ .

different approach directions of the platform relative to the object or different orientations of the object relative to the ground.

In some cases the projection of an object onto the image plane not only differs due to variations in the viewing direction, but also to varying geometric properties of the object itself. An example of this phenomenon is the rotating turret of a tank. The projection of a tank onto the image plane may be quite different directing the barrel forward or directing the barrel sideward. To model this effect, images are taken of tanks with different positions of the turret. The turret position is indicated by the angle  $\varphi$ . An angle of  $\varphi=0^\circ$  represents a tank with the barrel directed forward, where an angle of  $\varphi=+90^\circ$  represents a tank with the barrel directed to the right side, etc. Actual images are taken with barrel angles  $\varphi=0^\circ$ ,  $\varphi=15^\circ$ ,  $\varphi=30^\circ$  and  $\varphi=45^\circ$ . The dramatic effect of the turret rotation on the projected view of the object can be seen in Image 4.3. For a straightforward

vision system it is hardly possible to determine that these images do originate from one and the same object. A suggestion how to solve this problem is given in Section 5.

**Table 4.1** Image database overview. The figures within brackets represent the number of images per given turret angle and viewing direction.

label	vehicle type	turret	turret angle	viewing direction	number of database images
T1	Leopard 2 tank	yes	0°, 15°, 30°, 45°	0°, 30°, 60°	72 (6)
T2	M109 Howitzer	yes	0°, 15°, 30°, 45°	0°, 30°, 60°	72 (6)
T3	T-80 MBT	yes	0°, 15°, 30°, 45°	0°, 30°, 60°	72 (6)
T4	Gepard	yes	0°	0°, 30°, 60°	18 (6)
T5	M113 APC	no	0°	0°, 30°, 60°	18 (6)

## 4.2 Neural Network based Classifiers

In recent years neural networks have evolved into very powerful signal processing algorithms for all kinds of applications. In this section we will focus on the relation between pattern recognition and classification with respect to neural networks. A short overview of the basics of neural network computing is given where the emphasis lies on the error back-propagation network paradigm, as used in the multi-layer perceptron MLP network.

### 4.2.1 Neural Networks and Pattern Recognition

The application area of neural networks within the context of object classification is related to the field of pattern recognition. Pattern recognition can be considered as a process performing a classification operation on an given input. This implies that we have a predefined taxonomy of classes which is a formal or informal definition in what way members of a class are distinct from members of other classes. Note: there are networks that do define their own taxonomy, e.g. the Kohonen self organising feature maps [Kohonen, '88].

In general, the input for a pattern recognition system exists of a set of measurements comprising signal characteristics combined into a pattern or feature vector. A classification algorithm has to



determine for each input and for every class of patterns the probability that the input is a member of a particular class.

The classification is based on weighting features that are discriminating characteristics for the problem of interest. When the characteristic features are distinct for each class the classification is simple. More often however, characteristic features are not discriminating for all different classes. Moreover, in real world applications the pattern signatures are often buried in noise which make the specific characteristics even more diffuse. Therefore more than one or even a whole set of characteristics may be involved to define non ambiguous decision criteria.

Though its importance, the selection of a set of characteristics is often only the beginning to the solution of the classification problem. The next step lies in the generation of a practical description of the selected characteristics. This is the real problem since for many problems a concise formal and explicit description of the pattern characteristics is very hard or even impossible to give. The ability of neural networks to learn by example therefore is the key to the solution to circumvent the necessity of working with explicitly described characteristics. A neural network obtains knowledge about feature values common to members of a particular class and their corresponding tolerance regions by processing examples of different classes presented at the input of the network. This process is called training or learning in analogy of the behaviour of biological species.

The essence of the application of a neural network as a classifier lies in the network's ability to recognise patterns defined as input vectors and belonging to one particular class, as a cluster of points in a multidimensional feature space: the network can be trained to respond to each pattern belonging to a particular class by activating only one and the same output node. The output node then is assigned to this particular class. Hence, neural networks are capable to learn relevant classification characteristics by example instead of making use of formal descriptions.

#### 4.2.2 A Short Introduction to Neural Networks

A neural network is composed of many tightly interconnected non-linear processing elements or computational units. All the units operate in parallel. Each processing element belongs to a group that is called a layer and only units of different layers are interconnected. In general two types of layers are distinguished: layers that interact with the environment (input and output layer) and

layers that interact with other layers (hidden layer). By definition all units within one layer have the same functionality.

Each unit in the network computes a scalar output or activity level as a function of the input values to the unit. The instantaneous output values of the individual nodes together define the internal state of the network. This internal state can be regarded as a short-term working memory. Long-term storage is achieved by modifying the patterns of interconnection strengths among the units, i.e. by modifying the weights associated with each connection. Changes in the weight values are determined by learning rules adapting the unit's response, and hence the network output, to changes of the environment. These changes depend on the nature of the input signals and the desired output responses. In this way the network learns, i.e. organises information within itself. For an overview of neural network learning rules see [Lippmann, '87].

#### 4.2.3 Error Back-Propagation Neural Networks

The neural network paradigm that is considered for the classification problem belongs to the class of error back-propagation neural networks. Error back-propagation is a generally applicable rule to train a network and will be explained in Appendix A.

A typical back-propagation neural network consists of a three layer feed-forward network architecture. The three layers are generally referred to as input layer, hidden layer and output layer respectively. The input layer can be regarded as the fan-out of the input pattern and hence exists of a number of nodes that is equal to the dimension of the input vector. The input layer is fully connected to the hidden layer in the same way as the hidden layer is fully connected to the output layer. In some applications the input layer is also directly connected to the output layer but this option will not be considered. In Figure 4.2 an example of the topology of a three layer neural network is given. The interconnections between the nodes in successive layers are depicted schematically. The network's topology, the activation functions of the nodes and the interconnection strengths determine the input-output relation of the network.

The actual processing is done by the elements in the hidden layer and the output layer. Each processing element in the hidden layer receives one interconnection from each element of the input layer and each processing element of the output layer receives one interconnection from each element of the hidden layer. Associated with each of the interconnections is an adaptive weight  $w_{ij}$  where  $j$  refers to the originating node and  $i$  refers to the receiving node. In addition, each processing element in the hidden and output layer has one extra constant valued input and an

associated weight factor  $w_{iTh}$ . This weight is referred to as the *threshold* or bias and provides the network nodes with an extra degree of freedom, making the network more flexible to approximate a broader range of mappings. In general, the extra input value is set to one.

The output of a processing element is the result of applying an activation function to the weighted sum of the input values to that element. The activation function that is most commonly used is the non-linear sigmoid function. Other non-linear activation functions are described in [Shynk, '90]. A linear activation function is not considered here since the transformation abilities of a linear three layer network do not exceed those of a linear two layer network. The weighted sum  $s_i()$  and the activation function  $f()$  are given in Equation 4.1a and 4.1b.

$$s_i = \sum_{j=1}^{N_i} w_{ij} o_j + w_{iTh} \quad (4.1a)$$

$$o_i = f(s_i)$$

$$f(s_i) = 1 / (1 + \exp(-s_i)) \quad \text{if sigmoidal activation is used} \quad (4.1b)$$

Here  $N_i$  is the number of input connections to node  $i$  and  $o_j$  the output of node  $j$  in the previous layer. As mentioned before, the input layer can optionally be connected to the output layer. The weighted sum of Equation 4.1a for the output layer node then is extended by an extra term representing the weighted sum of the input layer output.

#### 4.2.4 Learning Rules

An error back-propagation network adapts the transformation function performing the optimum mapping from the input domain to the output domain by a process of learning by example. This requires that the desired network response for every distinct training input pattern must be known. Furthermore, requiring the transformation function to cover the entire output range, the input set should be an adequate representation of the whole input domain.

The neural network response to an input pattern is determined by the activation function and the weight factors assigned to each node. By adapting the weights the network response can be

changed. An optimum set of weight factors is that set for which the network responses to all training patterns match as close as possible the desired responses.

Once the network has learned all the transformations of the training sets, the network performance can be evaluated. New input vectors generate output vectors that are the result of an interpolation process of the mapping from the input domain to the output range. The encoding of complex decision boundaries by a limited number of parameters (weights) is due to the non-linearity of the transformation functions. Due to the non-linearity of the transformation functions, the network has the tendency to transform the new inputs in the same way as the trained input it is most similar to. This is the reason why neural networks can be successfully used in classification systems.

#### 4.3 Neural Networks and Related Techniques

The principles of distributed computing, adaptive networks and connectionism offer a promising framework to solve various difficult classification and related problems. However, there are also certain drawbacks and uncertainties related to these techniques.

The information in a neural network is stored in a distributed way in the strengths of the interconnections between the nodes. These weights are adapted during the learning phase. In this phase the network error function is minimised in a finite number of steps. This adaptation of the weights is a very precarious task since the information already stored in the network may not be lost. The settling of the network is therefore very time consuming (many iterations). Moreover, it can not be guaranteed whether the final stable state is a local or the global minimum of the network error function.

The so-called learning in adaptive neural networks is related to the fitting of data with hyperplanes in a multidimensional space. Following this relation, the mechanism of interpolation between known datapoints (input vectors) that the networks are expected to possess, becomes more explicit. In [Broomhead and Lowe, '88] a class of adaptive networks is presented that can be learned simply by solving a set of linear equations. Here, every input vector is represented by a set of radial basis functions. Hence, these networks have a learning rule, guaranteed to converge in a predefined number of iterations.

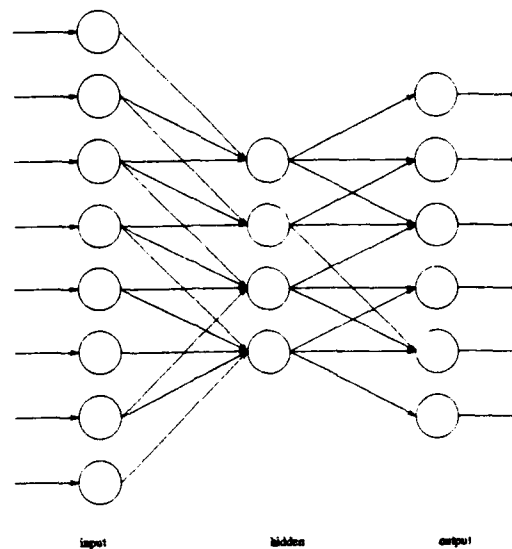


Figure 4.2 Schematic view of a three layer neural network comprising an input layer, a hidden layer and an output layer. The arrows represent interconnections with adaptable strengths.

Another class of networks that do not have a learning phase at all are the probabilistic neural networks [Specht, '90]. These networks incorporate a Bayes classifier where the decision surfaces converge to the Bayes-optimal boundaries when the number of training vectors increases.

Which of the three network types is optimal for the problem to be solved depends on constraints such as the available training data, computer power and learning time. In the next section more details about the last approach will be presented since the Bayes classifier based on a Parzen like PDF estimation is used as a reference to evaluate the neural network based classifier.

In Section 2 we have shown how we can select a restricted number of features that characterise a, possibly rotated, object in an image. Due to noise and discretization errors, the extracted Zernike moment parameters, though related to the same object type, are never exactly the same.

Therefore, we need a clustering mechanism that can separate feature vectors related to one object class, from feature vectors from other object classes. To implement this mechanism, we have selected the Multi-layer Perceptron Classifier, or MLP [Rumelhart et al., '86]. A MLP is a fully interconnected feed-forward neural network with one or more layers of nodes between the input and output layer of nodes.

As input to the network, we select the ordered set of Zernike moments. When the maximal order of moments used is 12, this means that the dimension of the input vector equals to 49, for a maximal order of 20 the number of moments equals to 121 (see Table 2.2 in Section 2).

The number of output nodes is equal to the number of object classes to be separated. To each output node one class is assigned. The network is trained to respond with all output nodes to be set to 0, except for the node that is marked to correspond to the class the input is from. That output node is set to 1.

There is one drawback in using MLP's. It has been shown that a MLP with at most two hidden layers can form any arbitrarily complex decision region in a feature space [Lippmann, '87]. However, there does not exist a specific rule for selecting the appropriate number of nodes in the hidden layer(s). The optimal network topology can only be determined by trial and error.

#### 4.4 Conventional Statistical Classifiers

Neural Networks can only be considered as an acceptable alternative to traditional classifiers if they perform at least as well as statistical classifiers or even outperform them. Therefore, to be able to draw some conclusions regarding the performance of neural networks, we have to compare them with other techniques. We have selected two well-known classifiers for comparison that will be evaluated in parallel: A Bayes classifier based on the Parzen estimator and a non-parametric nearest neighbour classifier. The same training vectors that are used to learn the neural network are used either to determine an optimal PDF function or to define a distance measure. Both approaches will briefly be discussed below.

### Bayes Classifier

A general approach towards classification problems is to design certain decision rules or strategies that minimise the expected risk or costs. Such strategies are called Bayes strategies [statistics], and are applicable to classification problems involving any number of classes. To be able to implement a Bayes decision rule, we must be able to calculate the probability density functions for the different classes, i.e. a set of functions describing the probability that a pattern belongs to each of the individual classes. Although in many cases the a priori probabilities are known or may be estimated accurately, in other situations a set of examples or training patterns  $P_C$  of the to be discriminated classes  $C_i$  are the only information that is available. Parzen [Parzen, '62] presented a set of probability density function estimators that provide an estimate of the underlying density provided that it is smooth and continuous. The estimator we selected is given by

$$f_{C_i}(P) = \frac{1}{(2\pi)^{N/2} \sigma^N} \frac{1}{m} \sum_{k=1}^m \exp \left[ - (P - P_{C_i^k})^T (P - P_{C_i^k}) / (2\sigma^2) \right] \quad (4.2)$$

Here  $f_C(P)$  is the PDF estimator for class  $C_i$ ,  $P$  the measurement vector, feature vector or evaluation vector and  $P_C$  the  $k$ th training vector of class  $C_i$ . The estimator uses the training patterns  $P_C$  as kernels i.e. the patterns are selected as centres for a set of multivariate Gaussian distributions. As can be seen from Equation 4.2 the probability density function is a sum of Gaussians scaled by certain factor.

Given the probability density functions for the classes involved, the Bayes decision rule assigning pattern  $P$  to class  $C_i$  is given by  $d_B(P) = C_i$  such that

$$h_{C_i} l_{C_i} f_{C_i}(P) \geq h_{C_j} l_{C_j} f_{C_j}(P), \quad i \neq j. \quad (4.3)$$

Here,  $f_C(P)$  represents the probability density function of a class  $C_i$ ,  $P$  is a  $N$ -dimensional input vector,  $l_C$  the loss associated with making the decision that  $P$  is a pattern of class  $C_i$  when the class is actually another one and  $h_C$  are the a priori probabilities of the occurrence of a pattern of class  $C_i$  respectively. The values of the losses are based on the consequences of making a incorrect decision. Assigning these values is part of the classification problem definition. For simplicity, in our experiments the values of the losses associated with making a correct decision are zero and the values of the losses associated with making an incorrect decision are all equally

set to 1. The width's of the Gaussians are controlled by the parameter  $\sigma$ . In our experiments,  $\sigma$  is set to 1.

#### Nearest Neighbour Classifier

Where the previous classifier is controlled by the parameter  $\sigma$ , the nearest neighbour classifier is a non-parametric classifier. To assign an input vector  $P$  to a certain class  $C_i$ , the nearest neighbour of  $P$  is determined among the set of all available training vectors. The class of this vector is assigned to the vector  $P$ . Hence, the unknown input vector  $P$  is assigned to class  $C_i$ , following  $d_{NN}(P) = C_i$ , where

$$i^* = \text{Min}_i d(P, P_{C_i}^k) \quad \forall i, \forall k, \quad (4.4)$$

with  $i$  ranging over all classes and  $k$  ranging over all members per class and  $d()$  an Euclidean distance measure.

### 4.5 Classification Results

#### feature vector normalisation

In our classification experiments we are working with object features grouped into feature vectors. Each feature represents a Zernike moment of a predefined order and repetition. In Figures 5.3, 5.4 and 5.5 several feature values are enlisted. From these figures it follows that different features have various dynamic ranges. Therefore, it is possible that a small group of features will predominate the characteristic pattern that is represented by the feature vector.

A neural network that is trained with these vectors may trigger on those dominant features only. To overcome this problem, i.e. to make sure that each feature will be equally weighted, the features must be normalised. The normalisation consists of the subtraction of the mean and the division by the standard deviation of the whole set of training samples. As a result, the training feature vectors have zero mean and unit variance before they are input to the network. The  $m$ th feature of the feature vector is normalised by

$$\bar{p}_m = \frac{p_m - \bar{p}_m}{\sigma_m} \quad (4.5)$$



where  $p_m$  and  $\sigma_m$  are the sample mean and standard deviation of the  $m/h$  training features of all the classes.

Since the vectors are also used in the reference classifiers, the vectors are also scaled to unit length, i.e. the sum of the squared vector entries is equal to 1.

## 5. EXPERIMENTS DESCRIPTION AND SIMULATION RESULTS

In this section the simulation results concerning the application of complex Zernike moments as rotation invariant object characterising features are presented. The emphasis of this section lies on the description of the experiments related to combination of the Zernike moments feature vectors and the neural network based classifier. In Section 5.1 the implementation of the automatic target recognition system as described in Section 1 is presented. This reduced scheme comprises only the feature extraction and classification module. In Section 5.2 the experiments are described and motivated and in Section 5.3, 5.4 and 5.5 the results are interpreted.

### 5.1 Automatic Target Recognition Scheme (Implementation)

To be able to perform experiments within a realistic context, we have implemented a part of the automatic target recognition processing scheme as depicted in Section 1. Actually, we only implemented the feature extraction module and the classification module. The preprocessing module and the object detection and segmentation modules are omitted because our main interest lies in the evaluation of the complex Zernike moments as rotation invariant object characterising features. We do realise, however, that those modules are of even greater importance and even more difficult to realise in an actual operational system. Consider, for example, the difficulties in extracting the silhouette object from an original grey-level image as is depicted in Image 4.1. This is a far from trivial problem, especially under changing light- and background-conditions. However, solving those problems is an image processing task and not a classification task.

The feature extraction module in our experiments exists of an image normalisation module and a complex Zernike moments extraction module. The image normalisation module is necessary to position the object of interest in the centre of the image plane and to scale the object to an uniform area. This preprocessing is necessary because the numerical values of the extracted Zernike features do depend on the size of the object within the imaginary unit circle superimposed onto the image plane..

To centre the object of interest, first the centre of mass of the object is determined. The centre of mass is obtained making use of Equation 2.3 of Section 2.2. The object is then translated and scaled such that the centre of mass lies in the centre of the image array and the area occupied by the object has a predefined value (= number of pixels). The translation and scaling of an object in the

image plane is depicted in Image 5.1. In this image also the imaginary unit circle superimposed onto the image array is shown.

The second module we have implemented is the Zernike rotation invariant features extraction module. This module is entirely based on the equations presented in Section 2 and the software modules presented in Section 3. The module makes use of the fast implementation based on the Zernike moment templates.

We have already mentioned the imaginary unit circle superimposed onto the image plane. This unit circle is important since the orthogonal basis functions of Equation 2.4, on which the Zernike moments are based, are only defined within this region. The unit circle makes it possible to extract Zernike moment coefficients from an image independent of the image size by applying an appropriate scaling factor evaluating Equation 2.7 which is based on the area (in number of pixels) of the unit circle. As a result of this scaling, the Zernike moments extracted from the images having different sizes are almost the same. The image sizes vary from 64x64, 128x128 to 256x256 pixels. Small variations in the moment coefficients are due to roundoff errors in resizing the image plane. The resulting Zernike moments of these images are depicted in Figure 5.1.

For the classification experiments described in the next subsection the Zernike moments up to order 20 are determined in advance for all images in the image database. The absolute values of the Zernike moments are placed in a feature vector of dimension 121 (order  $n=[0-20]$ , for all valid repetitions  $m$ ). This vector or parts of this vector will be used as an input pattern by the different classifiers evaluated in the experiments as described in the next subsection.

## 5.2 Classification Experiments within the RPV Monitor Context

Within the context of the project as described in Section 1, we have set up three types of experiments. First, we want to find out what the performances are of neural network based classifiers in comparison with traditional statistical classifiers. Second, we are interested in the characteristics of the Zernike complex moments as rotation invariant features for automatic target recognition. More specifically, we want to find out what the relation is between the dimension of the input vector, i.e. the number of moments used to characterise an object, and the classification results obtained by the various classifiers. Considering neural networks, we are also interested in the relation between the number of hidden nodes in the neural network and the classification result. Third, we want to find out what the generalisation capabilities are of the various classifiers with

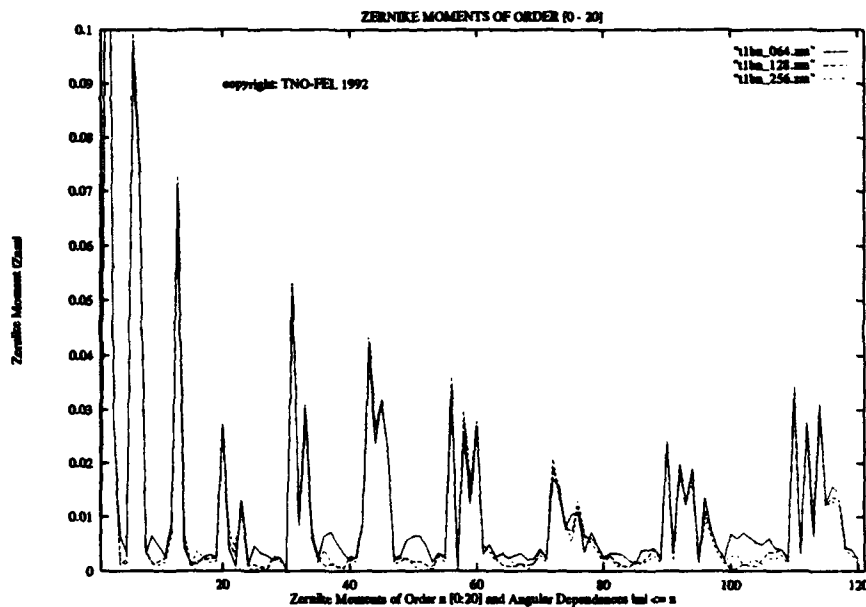


Figure 5.1. Absolute values of the first 20 Zernike moments (=121 moments for all valid repetitions) extracted from identical image, only varying in size (64x64, 128x128, 256x256).

respect to changes in the viewing direction relative to the objects in the images. As we have seen in Section 4, the appearances of objects can differ dramatically when the viewing direction is changed. This effect must be taken into account designing a reliable classifier.

The neural network type we have investigated is the well-known Multi-layer Perceptron MLP error back-propagation neural network type as described in Section 4.2. For a reference, we selected the nearest neighbour classifier and the Parzen estimator [Parzen, '62] as representatives of the traditional statistical classifiers.

As we mentioned in Section 4, the total number of different classes the classifier has to discriminate, equals five. Three of the classes may be subdivided into subclasses, characterised by their relative turret position. However, in the experiments described, all subclasses are mapped onto

one and the same main class. In Image 5.2 top view images of all five objects are depicted. As a reminder, in Table 4.1 in Section 4, an overview of the image database is listed.

For all (sub)classes, images taken from different viewing angles are available. However, no distinctions are made between different viewing angles during evaluation of the results in the first two experiments.

The evaluation of the classifiers' output is based on determining the number of correct classifications relative to the total number of evaluation inputs. This means that no attention is paid to the number of incorrect classifications.

This may blur the interpretation of the classification statistics, since a high classification score, in combination with a relative score of mis-classifications may in some situations be interpreted worse than a lower classification score in conjunction with a false alarm rate of zero.

Nevertheless, we have decided not to take into account the false alarm rate, since this biased the results in a way such that the characteristics of the various classifiers may be obscured (remember that our main point of interest lies in the evaluation of the performance of Complex Zernike Moments as invariant features, not to develop an optimal classifier strategy).

For the MLP and the Parzen estimator, the class label corresponding to the output node with the highest output value is assigned to the input vector. For the NN classifier, the class label of the output having the lowest output value is assigned to the input vector. For those classes, having subclasses, we made use of 24 training vectors per class to train the neural network. For the 2 classes without subclasses, we made use of 6 training vectors per class. The same vectors are used as a reference database for the NN and Parzen classifier. The number of evaluation input vectors for the classes with subclasses was equal to 48 where the number of evaluation vectors for the classes without subclasses was equal to 12.

### 5.3 Performance comparison between different classifiers

The results of the comparison of the Multi-Layer Perceptron neural network, the nearest neighbour classifier and the Parzen classifier are given in Figure 5.1a until 5.1d. In each graph the relation between the number of features in the input vector and the classification accuracy is depicted. From the successive graphs the influence of the signal to noise ratio on the classification results can be deduced.

In Figure 5.5a, the classification results are depicted based on input vectors extracted from images without addition of noise. In Figure 5.5b, the contours of the objects in the input images were first corrupted by a noise process, resulting in a SNR of 36 dB. In Figure 5.5c and 5.5d results on images with a SNR of 25 dB and 20 dB are presented. In Image 5.3, the effect of this noise process on the object boundaries is depicted. As can be seen, the extraction of the correct object contour becomes more difficult when the SNR is decreasing. The effect of the image noise on the extracted Zernike moments can be seen in Figure 5.4.

Looking at the individual graphs, the results of the three classifiers do not differ dramatically. In general, the Parzen classifier performs worse for high signal to noise ratio's for all dimensions of the input vectors. The performance of the MLP neural network and the NN classifier are equally well at all SNR's for the highest input vector dimensions. The performance of all classifiers are going down with decreasing SNR's. It is known that the performance may be improved when noise corrupted input vectors are also used as training vectors or reference vectors. However, for reasons stated before, we did not take these into account.

A second conclusion we may draw from these graphs is that the total number of features taken into account in the classification process, has a positive effect on the final classification results. This can be seen from the downwards slope of the lines in the graphs: the higher the number of features in the input vector, the higher the classification accuracy.

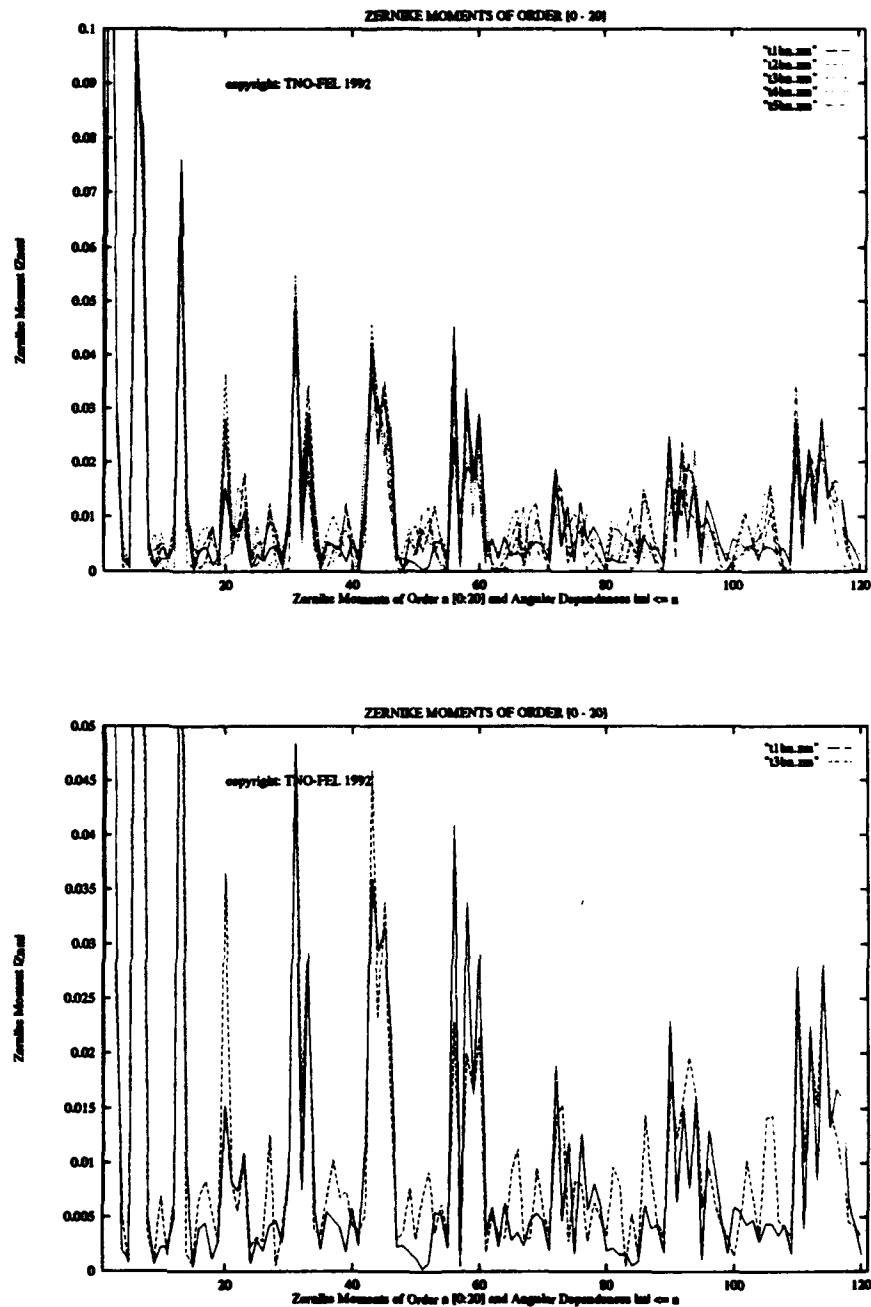


Figure 5.2. Absolute values of the first 20 Zernike moments (=121 moments for all valid repetitions): (a) moments of the 5 different objects in the database; (b) details of object t1 and t3;

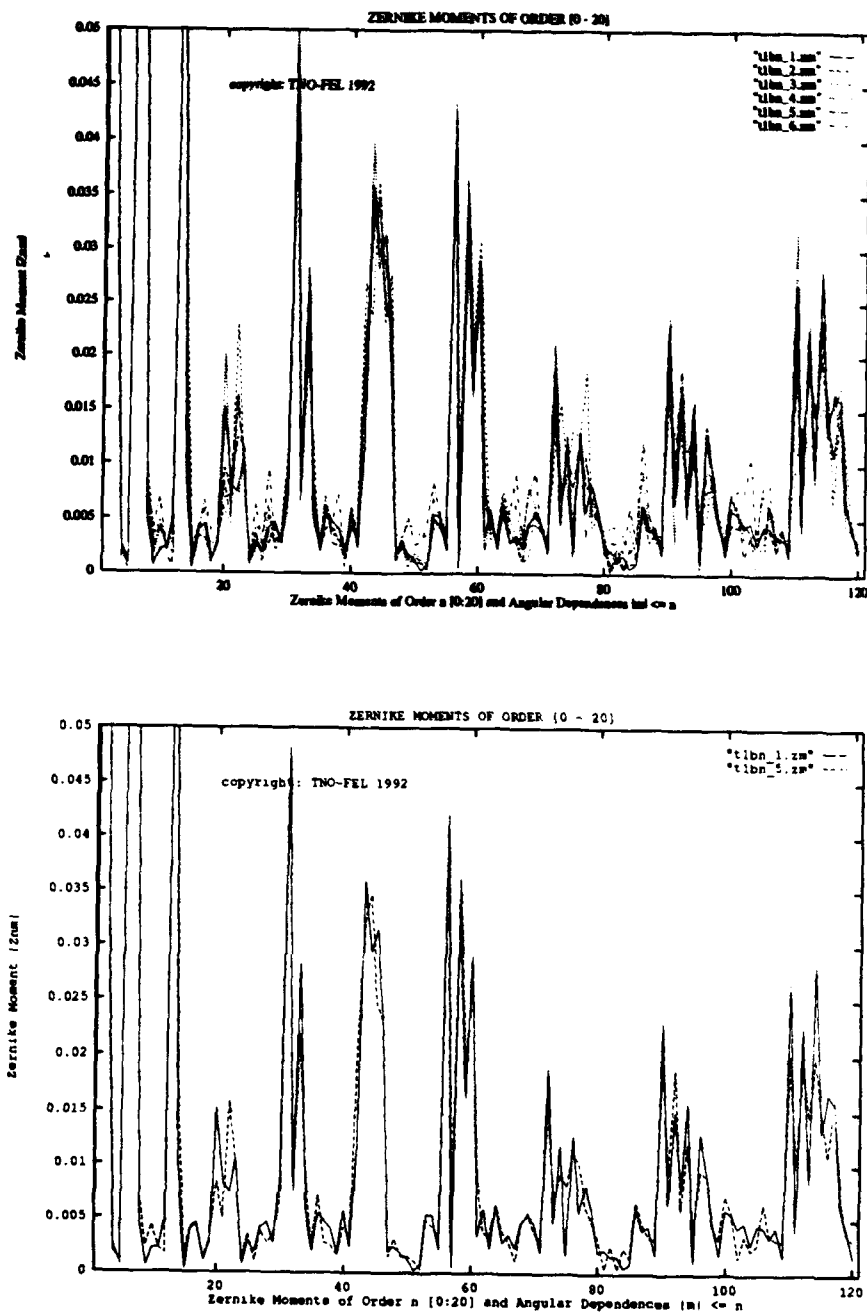


Figure 5.3. Absolute values of the first 20 Zernike moments (=121 moments for all valid repetitions): (a) moments of the 6 different object representations presented in Image 4.2; (b) details of image objects *a* and *e*;



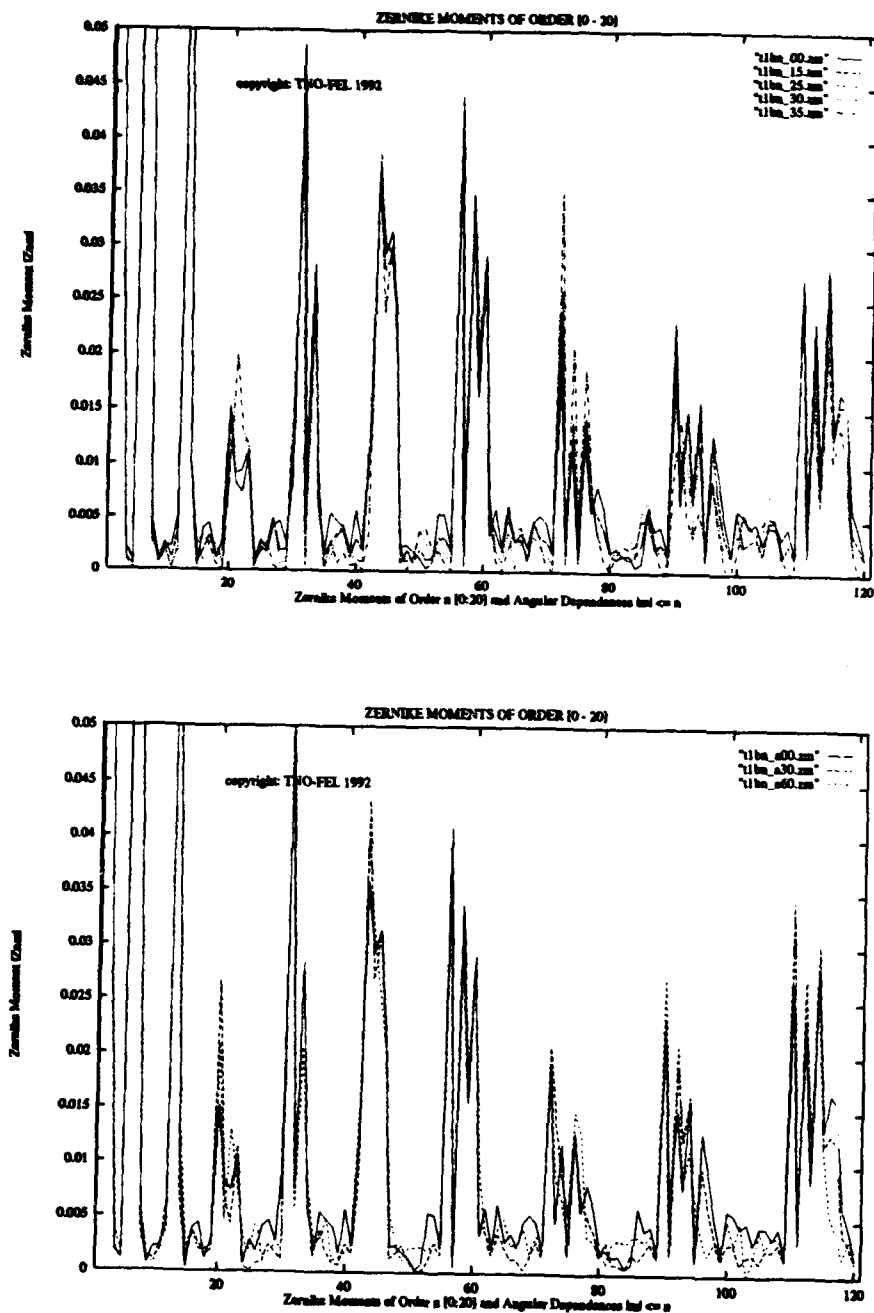
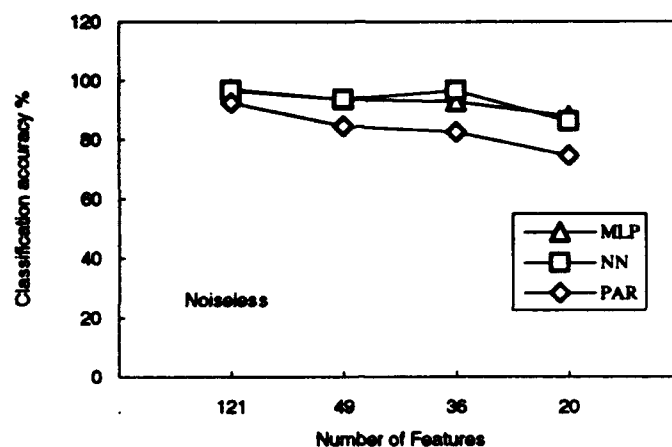


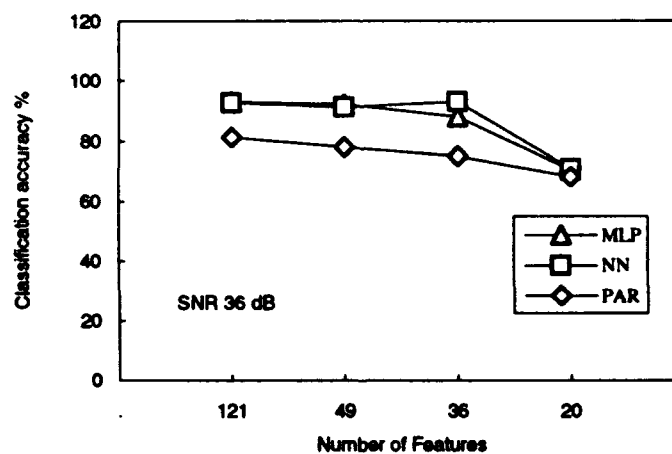
Figure 5.4. Influence of image noise (a) and viewing direction (b) on extracted Zernike moments: Absolute values of the first 20 Zernike moments (=121 moments for all valid repetitions)



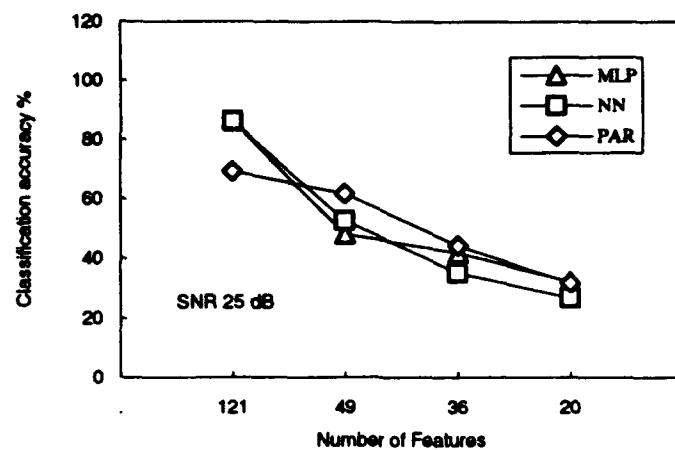
(a)

Figure 5.5

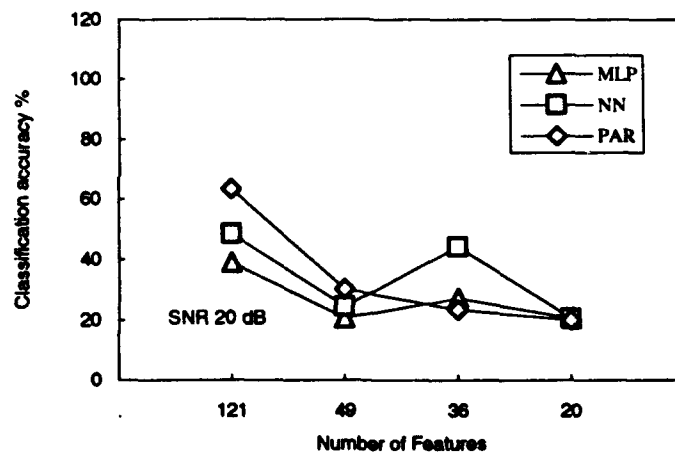
Classification results related to the number of features in the input vector using Zernike Complex moments as invariant features. Here, MLP, NN and PAR stand for Multi-Layer Perceptron neural network, Nearest Neighbour classifier and Parzan classifier respectively. The neural network has 50 nodes in the hidden layer. There are 5 classes and 84 input vectors for training and 168 input vectors for evaluation. (a) Noiseless; (b) SNR: 36 dB; (c) SNR 25 dB; (d) SNR 20dB;



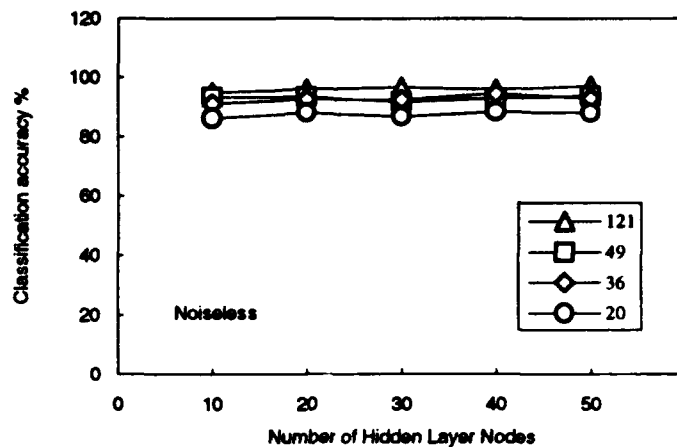
(b)



(c)



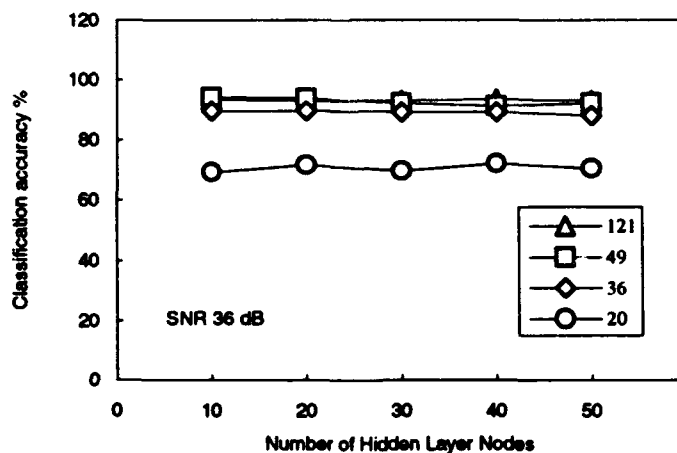
(d)



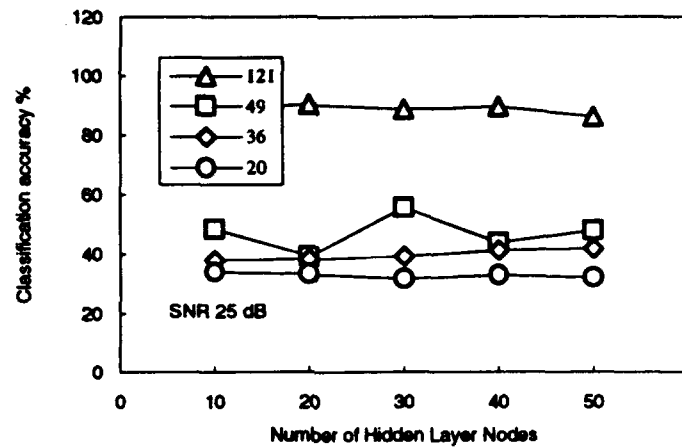
(a)

Figure 5.6

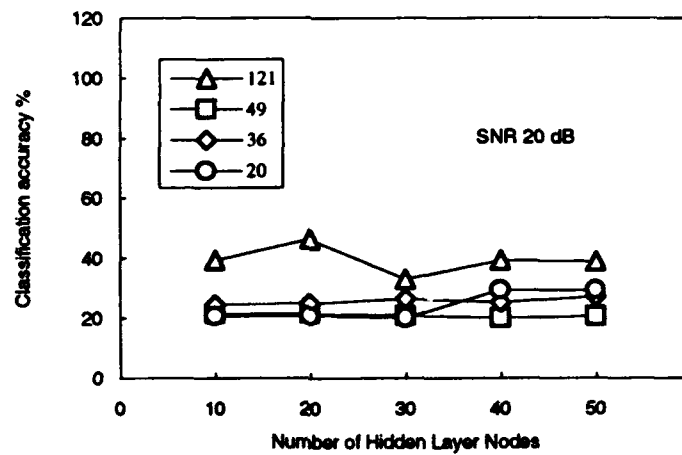
Classification results of Multi Layer Perceptron neural networks related to the number of hidden layer nodes, using Zernike Complex moments as invariant features. The neural networks vary in the number of features stored in the input vectors, ranging from 121, 49, 36, to 20, respectively. There are 5 classes and 84 input vectors for training and 168 input vectors for evaluation. (a) Noiseless; (b) SNR: 36 dB; (c) SNR 25 dB; (d) SNR 20dB;



(b)



(c)



(d)

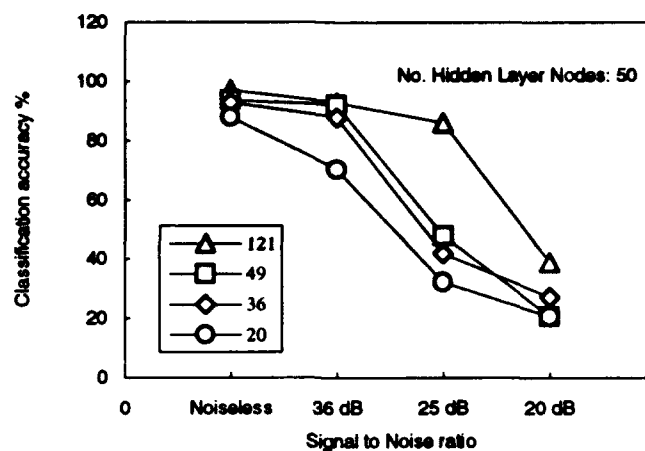
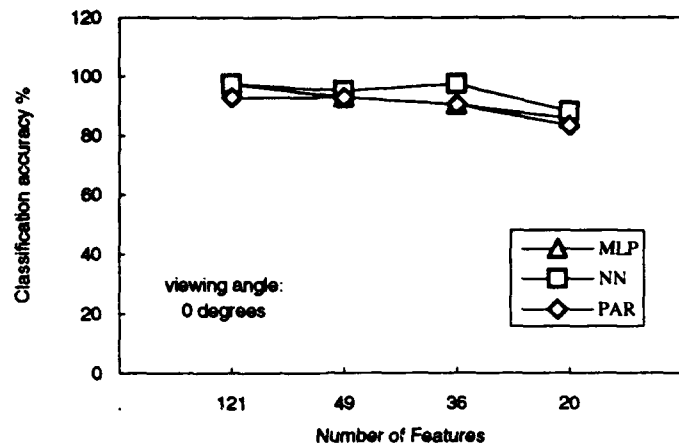


Figure 5.7

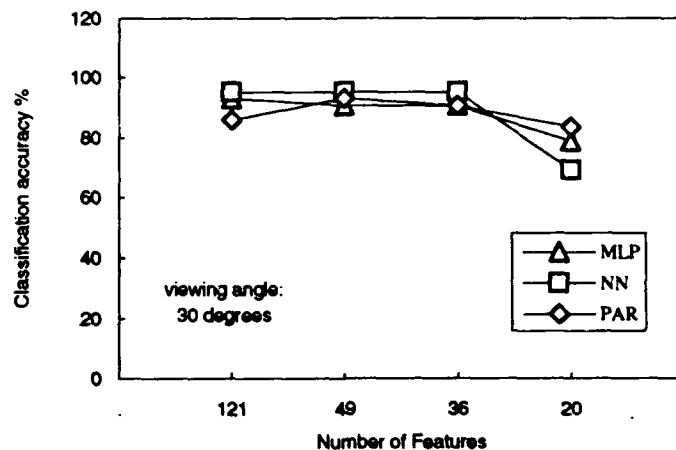
Classification results of Multi Layer Perceptron neural networks related to the signal to noise ratio of the images the Zernike Complex moments, used as invariant features, are extracted from. The number of nodes in the hidden layer equals to 50. The neural networks vary in the number of features stored in the input vectors, ranging from 121, 49, 36, to 20, respectively. There are 5 classes and 84 input vectors for training and 168 input vectors for evaluation.



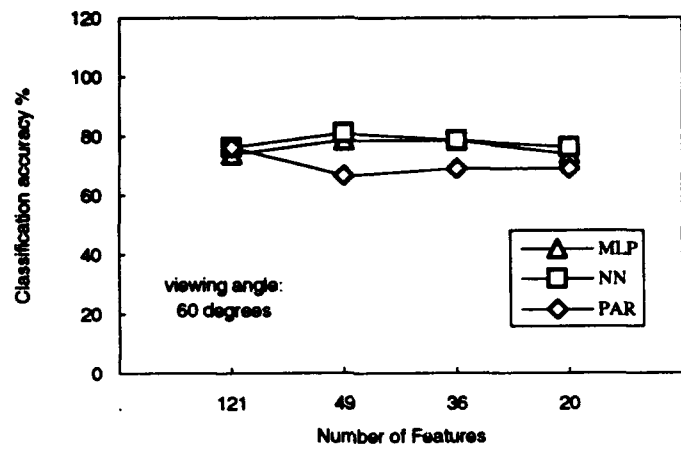
(a)

Figure 5.8

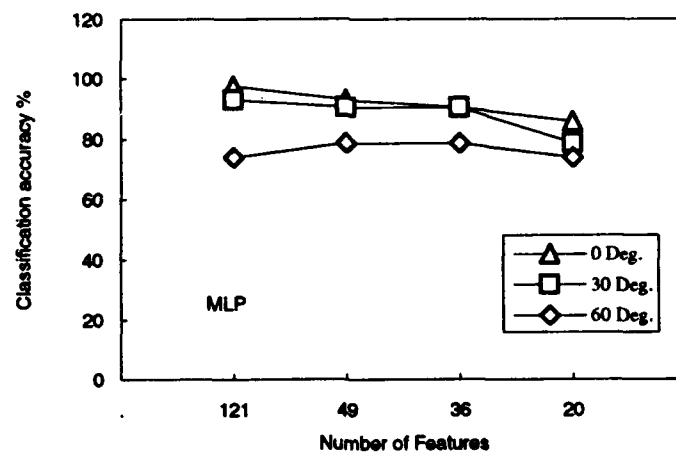
Classification results related to the viewing direction of the training data set and the evaluation data set. Here, MLP, NN and PAR stand for Multi-Layer Perceptron neural network, Nearest Neighbour classifier and Parzan classifier respectively. The neural network has 40 nodes in the hidden layer. There are 5 classes and 56 input vectors for training and 42 input vectors for evaluation. (a) Viewing direction 0 degrees; (b) Viewing direction 30 degrees; Viewing direction 60 degrees; (d) Viewing direction: all;



(b)



(c)



(d)



#### 5.4 Zernike Moments and Neural networks

Considering multi-layer perceptron neural networks, there are several network parameters that can be optimised, tuning the classifier. Focusing on the network topology, we may alter the number of hidden layers and the number of nodes in each layer. Since any mapping from the input space to the output space can be realised by a network having only one hidden layer, we did decide not to evaluate networks with more than one hidden layer. However, we did vary the number of nodes in this layer, while at the same time varying the dimension of the input vector, i.e. the number of Zernike features characterising the object.

In Figure 5.6 the results of the experiments related to the tuning of the network topology are depicted. The data sets are the same as described at the beginning of this section. In Figure 5.6a, the influence of the number of hidden nodes on the classification accuracy is depicted for 4 different input vector dimensions. It is striking that the dimension of the hidden layer hardly has any effect on the classification accuracy. However, the conclusions drawn from Figure 5.5 are again valid. The larger the dimension of the input vector, the higher the classification accuracy.

In Figure 5.6b-5.6d, the influence of the number of hidden layer nodes on the classification accuracy is depicted for various signal to noise ratio's of the input image. Again, the relatively small influence of this parameter on the classification accuracy is evident. On the other hand, the positive influence of the input vector dimension on the overall result is again clearly seen, especially in Figure 5.6c. Up to a signal to noise ratio of 25 dB, the classification accuracy, for a input vector dimension of 121, lies at an acceptable level of above 85%. For lower input dimensions, the accuracy is already unacceptable low. At a SNR of 36 dB, the results for an input vector dimension of 20 features is below 80%.

The positive effect of the input vector dimension is summarised in Figure 5.7. Since the number of hidden layer nodes hardly has any influence, this graph is only given for a number of hidden layer nodes equal to 50. Again, the robustness of the classifier for variations in the signal to noise ratio for a large number of input features can clearly be seen.

#### 5.5 Generalisation

The third series of experiments introduced in Section 5.1, are focused on gaining insight in the generalising capabilities of the various types of classifiers. Here, with generalisation we mean the

capability of a classifier to label a feature vector signature with the correct class label also in those cases where the signature is not lying within the range covered by the training vector set.

We have simulated this behaviour by training the neural network with vectors originating from objects monitored with viewing directions of 0 degrees and evaluating the neural network classifier with vectors monitored with viewing directions of 30 degrees and 60 degrees. As can be seen from Image 5.5 the projection of an object onto the image plane do changes dramatically with changing the viewing direction. Therefore, it may happen that an object viewed from one direction is more similar to another object than to the same object viewed from a different direction.

In Figure 5.8 the results of the generalisation experiments are depicted. As a reference, in Figure 5.8a the results for the various classifier types are presented. The classifiers are trained and evaluated with feature vectors originating from objects monitored from the same viewing direction of 0 degrees.

In Figure 5.8b the classification results are depicted based on a training set of 0 degrees and a evaluation set of 30 degrees. In Figure 5.8c the same results are given for a training set of 0 degrees and a evaluation set of 60 degrees. Finally in Figure 5.8d an overview of the neural network performance for the different viewing directions is given.

The generalisation capabilities are quite remarkable for the 30 degree case, especially for larger input dimensions. However, for the 60 degree case, the results are almost all below the 80%. This can be expected giving the variations in the projected images originating from different viewing directions as is depicted in Image 5.5. When the number of classes is extended, the results will probably be even worse. Nevertheless, all classifier types do show some degree of generalisation which again demonstrates the robustness of the Zernike moments for capturing object characteristics.

## 6. CONCLUSIONS, SUGGESTIONS FOR FURTHER RESEARCH AND CONCLUDING REMARKS

### 6.1 Conclusions

From the experiments described and evaluated in Section 5, several conclusions can be drawn related to the behaviour of Zernike moments in combination with neural networks in a classification scheme. However, given the limited scope of the project and the experiments set-up these conclusions are not in general valid and can not be extended to an actual operational classification system. The Zernike moments provide only a solution to the feature extraction and classification stages of a object recognition scheme. The object detection and segmentation modules are not covered by this research. Nevertheless, the experiments did give us clear insight in the behaviour and performances of the complex Zernike moments as rotation invariant object characterising features.

#### effectiveness Zernike moments

The complex Zernike moments have shown to be a very effective way to characterise objects for a scale, translation and rotation invariant automatic target recognition system. A classification score of up to 90 percent has been accomplished given a database with five different military vehicles monitored under different levels of scaling, translation and rotation in the image plane. Moreover, the coefficients are robust under scaling of the image plane dimension. However, the classification results are decreased considerably when identical objects are to be classified viewed from different viewing angles. This problem may be overcome when for each object training examples are available for all appropriate viewing directions. In short, the moments are robust under scaling, translation, rotation and noise and to a certain extend to varying affine projections.

#### complexity Zernike moments

The Zernike moments have shown to be a computationally complex approach towards the problem of rotation invariant object recognition. The computational overhead can be reduced by calculating in advance the Zernike polynomial coefficients. In this way run-time computation time can be

substituted by a large memory comprising the Zernike templates: computer power is substituted by computer storage.

On the other hand, the Zernike moments are efficient in a way that the contour characteristics are captured in a small number of coefficients relative to the contour complexity. The relative insignificant influence of the number of coefficients on the classification results suggests that the dominant contour characteristics are captured in the first 10 to 20 coefficients.

#### Multi-layer perceptron neural network

The multi-layer perceptron neural network proved to be a robust classification system even under severe signal to noise conditions. However, the MLP neural network did not outperform in any way the straightforward classical nearest neighbour classifier. Though, the advantage of a neural network in an operational system can be that the classification time can be reduced. The extended example database that is used by the nearest neighbour classifier and the Parzen classifier is in the neural network case encoded in the weight coefficients of the network which is in general very efficient. The time to evaluate a given input feature vector can therefore be reduced considerably.

### 6.2 Suggestions for further Research and Concluding Remarks

Since the Zernike moments have shown to be an effective solution towards the problem of rotation invariant object recognition, further research in the field of moments in relation with an automatic target recognition system is appropriate.

In our experiments we only considered the last two stages of the object recognition scheme depicted in Section 1: feature extraction and classification modules. Further work may include the development of robust modules for the first two sections, i.e. the object detection and object segmentation modules. Especially, the object segmentation will be a serious problem in its own.

Furthermore, the extraction of the Zernike moments is a time consuming operation. Research in finding efficient real-time implementations is required. A suggestion to this problem is given in Section 3. Moreover, a study comparing the characteristics of the complex Zernike moments and other moments mentioned in Section 2 will give insight in the various alternatives and make a qualitative selection of one of the moments types to be implemented in a demonstration classification system possible.

Finally, as a second alternative to the automatic target recognition problem, model based object recognition may be considered. Interesting results have been reported in literature. 3 Dimensional models or a priori knowledge about the objects of interests may reduce the sensitivity of the classification system for errors in the segmentation process, still one of the most difficult problems to be solved in image processing.

## REFERENCES

- [Abu-Mostafa and Psaltis, '84] Abu-Mostafa, Y.S., and Psaltis, D., "Recognitive Aspects of Moment Invariants", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 6, pp. 698-706, November 1984.
- [Abu-Mostafa and Psaltis, '85] Abu-Mostafa, Y.S., and Psaltis, D., "Image Normalization by Complex Moments", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 7, pp. 46-55, January 1985.
- [Arbter, '89] Arbter, K., "Affine-invariant Fourier Descriptors", in *From Pixels to Features*, J.C. Simon (Ed.), North-Holland, Amsterdam, 1989.
- [Arbter et al., '90] Arbter, K., Snyder, W.E., Burkhardt, H., and Hirzinger, G., "Applications of Affine-invariant Fourier Descriptors to Recognition of 3-D Objects", *IEEE Trans. Pattern Anal. Machine Intell.*, PAMI-12, pp. 640-647, 1990.
- [Beck, '92] Beck, W., Applications of Motion Estimation in Image Sequence Processing, TNO-FEL report, FEL-92-B356, 1992.
- [Born and Wolf, '75] Born, M., and Wolf, E., *Principles of Optics*, Pergamon Press, Oxford, 1975.
- [Boyce and Hossack, '83] Boyce, J.F., and Hossack, W.J., "Moment Invariants for Pattern Recognition", *Pattern Recognition Lett.*, 1(5-6): pp. 451-456, July 1983.
- [Broomhead and Lowe, '88] Broomhead, D.S. and Lowe, D., "Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks", Memorandum 4148, Royal Signal and Radar Establishment, United Kingdom 1988.
- [Duda and Hart, '73] Duda, Richard O., and Hart, Peter E., *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1973.

[Dudani et al., '77] Dudani, S.A., Breeding, K.J., and McGhee, R.B., "Aircraft Identification by Moment Invariants", *IEEE Trans. Comp.*, C-26, 39-46, 1977.

[Gonzalez and Wintz, '77] Gonzalez, R., and Wintz, P., *Digital Image Processing*, Reading, MA: Addison-Wesley, 1977.

[Grace and Spann, '91] Grace, A.E., and Spann, M., "A Comparison between Fourier-Mellin Descriptors and Moment Based Features for Invariant Object Recognition Using Neural Networks", *Pattern Rec. Letters* 12, pp. 635-643, 1991.

[Hooton and Munson, '92] Hooton, E.R., and Kenneth Munson, eds. "Jane's Battlefield Surveillance Systems 1992", 1992.

[Hu, '62] Hu, M.K., "Visual Pattern Recognition by Moment Invariants", *IRE Trans. Inform. Theory*, IT-8, pp. 179-187, Feb 1962.

[Khotanzad and Hong, '90] Khotanzad, A., and Hong, Y.H., "Invariant Image Recognition by Zernike Moments", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 12, No. 5, May 1990.

[Khotanzad and Lu, '90] Khotanzad, A., and Lu, J-H., "Classification of Invariant Image Representations Using a Neural Network", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol.38, No. 6, June 1990.

[Kohonen, '88] Kohonen, T., *Self-Organization and Associative Memory*, Springer-Verlag, Berlin, 1988.

[Lippmann, '87] Lippmann, Richard P., "An Introduction to Computing with Neural Nets", *IEEE ASSP Magazine*, Vol. 4, pp. 4-22, April 1987.

[Parker, '91] Parker, J.R., "Gray Level Thresholding in Badly Illuminated Images", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 13, No. 8, August 1991.

[Parzen, '62] Parzen, E., "On Estimation of a Probability Density Function and Mode", *Ann. Math. Stat.*, Vol. 33, pp. 1065-1076, 1962.

[Perantonis and Lisboa, '92] Perantonis, S.J., and Lisboa, P.J.G., "Translation, Rotation, and Scale Invariant Pattern Recognition by High-Order Neural Networks and Moment Classifiers", *IEEE Trans. Neural Networks*, Vol. 3, No.2, March 1992.

[Persoon and Fu, '77] Persoon, E., and Fu, K.S., "Shape Discrimination using Fourier Descriptors", *IEEE trans. Syst. Man. Cybernetics*, SMC-7, pp. 170-179, 1977.

[Prokop and Reeves, '92] Prokop, R.J., and Reeves, A.P., "A Survey of Moment-Based Techniques for Unoccluded Object Representation and Recognition", *CVGIP: Graphical Models and Image Processing*, Vol. 54, No. 5, pp. 438-460, September 1992.

[Reddi, '81] Reddi, S.S., "Radial and Angular Moment Invariants for Image Identification", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. PAMI-3, pp. 240-242, March 1981.

[Reiss, '91] Reiss, T.H., "The Revised Fundamental Theorem of Moment Invariants", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 13, No. 8, pp. 830-834, August 1991.

[Rosenfeld and Kak, '82] Rosenfeld, A., and Kak, A.C., *Digital Picture Processing*, 2nd Ed., Vol. 2, Academic Press, New York, 1982.

[Roth, '90] Roth, M.A., "Survey of Neural Network Technology for Automatic Target Recognition", *IEEE Trans. Neural Networks*, Vol. 1, No. 1, March 1990.

[Rumelhart et al., '86] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., "Learning Internal Representations by Error Propagation", in Rumelhart, D.E., and McClelland, J.L., editors, *Parallel Distributed Processing*, pp. 318-364, The MIT Press, Cambridge, Massachusetts, London, England, 1986.

[Schafer, '76] Schafer, G.A., *A Mathematical Theory of Evidence*, Princeton Univ. Press, 1976.

[Schalkoff, '92] Schalkoff, R., "Pattern Recognition, Statistical, Structural and Neural Approaches", John Wiley & Sons, Inc., 1992.



[Schau, '92] Schau, H.C., "Shape Recognition with Scale and Rotation Invariance", *Opt. Eng.*, Vol. 31, No. 2, February 1992.

[Sheng and Arsenault, '86] Sheng, Y., and Arsenault, H.H., "Experiments on Pattern Recognition using Invariant Fourier-Mellin Descriptors", *J. Opt. Soc. Am.*, A6, pp. 771-776, 1986.

[Sheng and Duvernoy, '86] Sheng, Y., and Duvernoy, J., "Circular-Fourier-Radial-Mellin Transform Descriptors for Pattern Recognition", *J. Opt. Soc. Am.*, A3, pp. 885-888, 1986.

[Shynk, '90] Shynk, John J., "Performance Surfaces of a Single-Layer Perceptron", *IEEE trans. on Neural Networks*, Vol. 1, No. 3, September 1990.

[SunVision, '91] SunVision 1.1: Reference Manual, Sun Microsystems, Inc., Mountain View, CA, 1991.

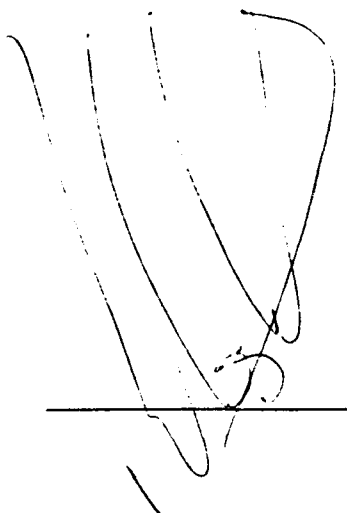
[Specht, '90] Specht, D.F., "Probabilistic Neural Networks and the Polynomial Adaline as Complementary Techniques for Classification", *IEEE Trans. Neural Networks*, Vol. 1, No. 1, March 1990.

[Teague, '80] Teague, M.R., "Image Analysis via the General Theory of Moments", *J. Opt. Soc. Am.*, Vol.70, No. 8, pp. 920-930, August 1980.

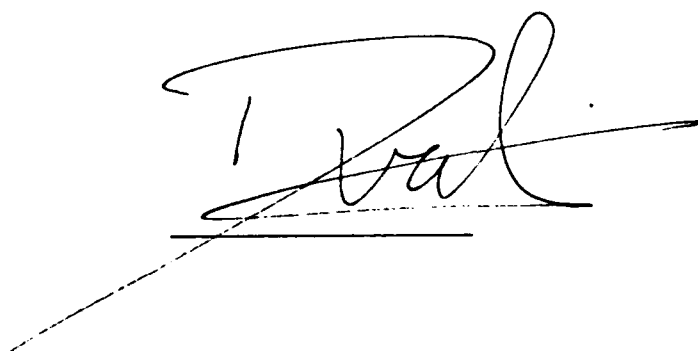
[Teh and Chin, '88] Teh, Cho Huak, and Chin, Ronald T., "On Image Analysis by the Methods of Moments", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.10(4), pp. 496-513, July 1988.

[Toet, '92] Toet, A., "Invariant Representations for Flat Image Forms: A Literature Survey" (in Dutch), TNO report, IZF 1992 A-16, 1992.

[Zernike, '34] Zernike, F., *Physica*, Vol. 1, p. 689, 1934.

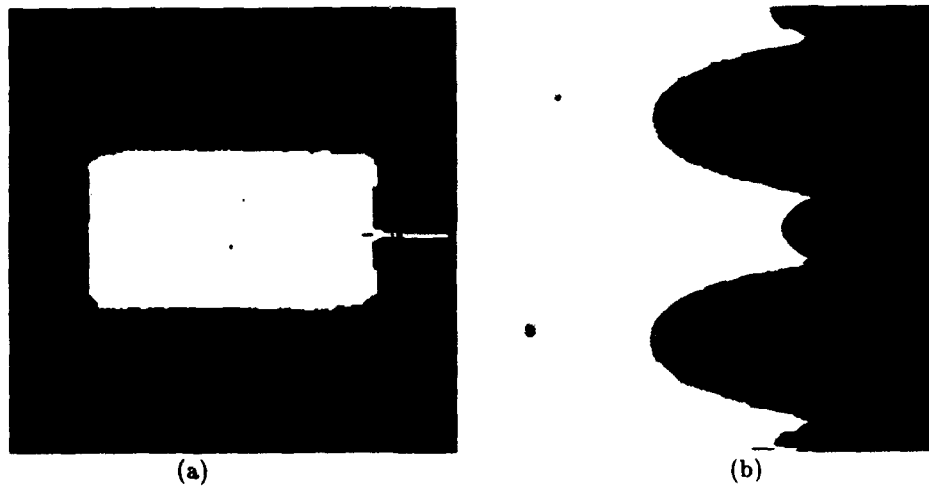
A handwritten signature in black ink, consisting of several overlapping loops and a long horizontal stroke at the bottom.

P.L.J. van Lieshout  
(Group leader)

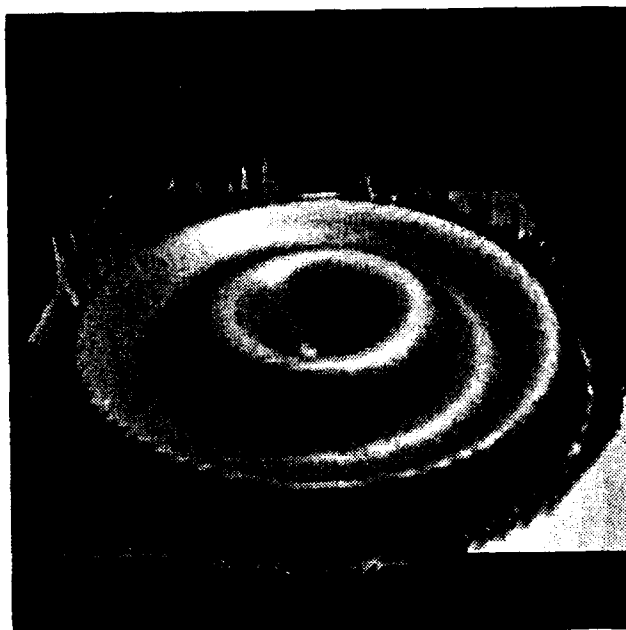
A handwritten signature in black ink, featuring a large, stylized 'K' and a long horizontal stroke at the bottom.

P.F.C. Krekel  
(Author)

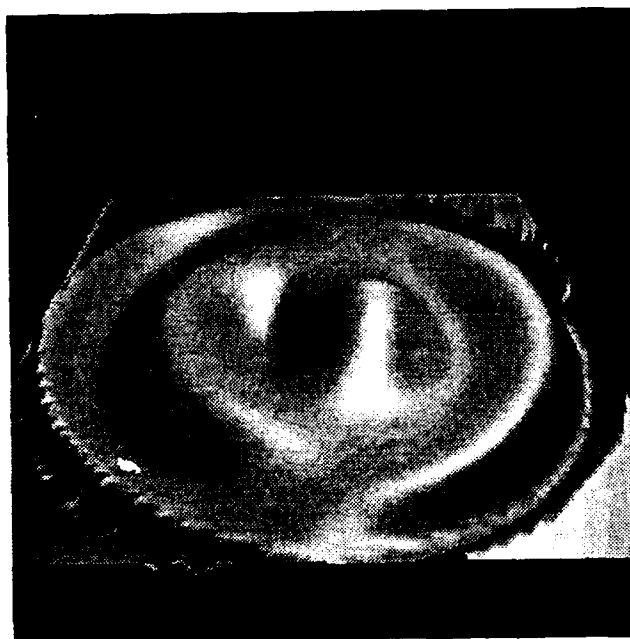
## IMAGES



**Image 3.1** Transformation of coordinate system from Cartesian to polar: (a) Cartesian coordinates; (b) polar coordinates;



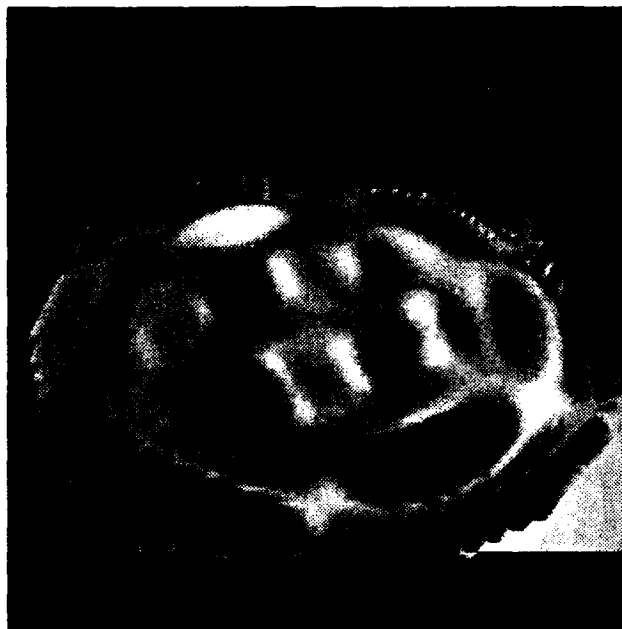
(a)



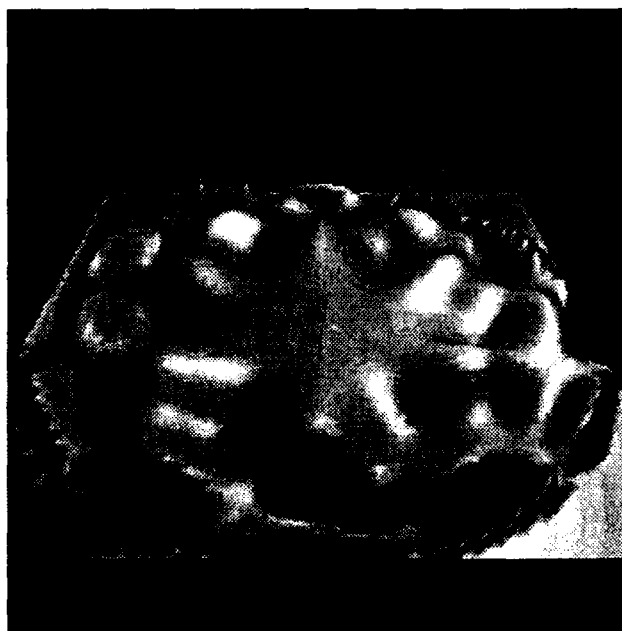
(b)

Image 3.2

Zernike complex moments templates: (a) template Z80 (order 8, repetition 0); (b) template Z81 (order 8, repetition 1);



(a)



(b)

Image 3.3

Zernike complex moments templates: (a) template Z83 (order 8, repetition 3); (b) template Z86 (order 8, repetition 6);

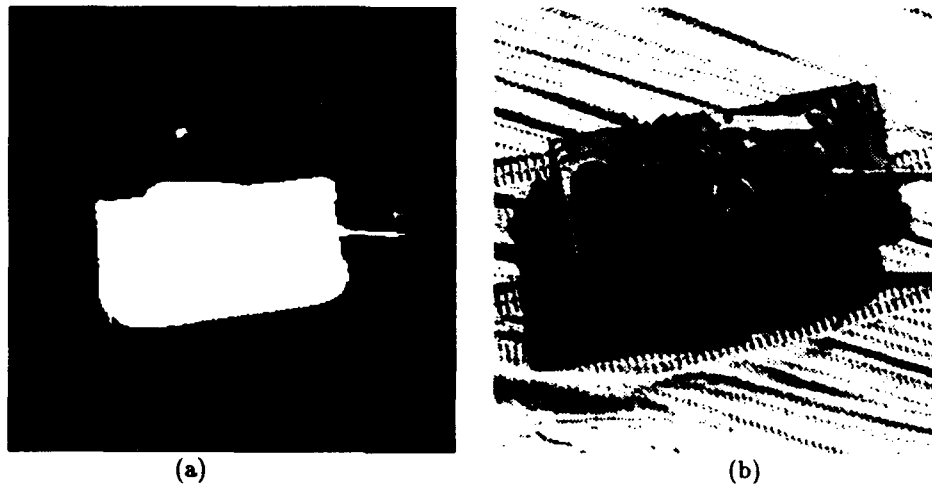


Image 4.1

Database example. The significance of the test image with respect to the real-life object can clearly be seen; (a) database example; (b) real-life image; Image (b) reprinted with permission of DMKL/DCAWACO.

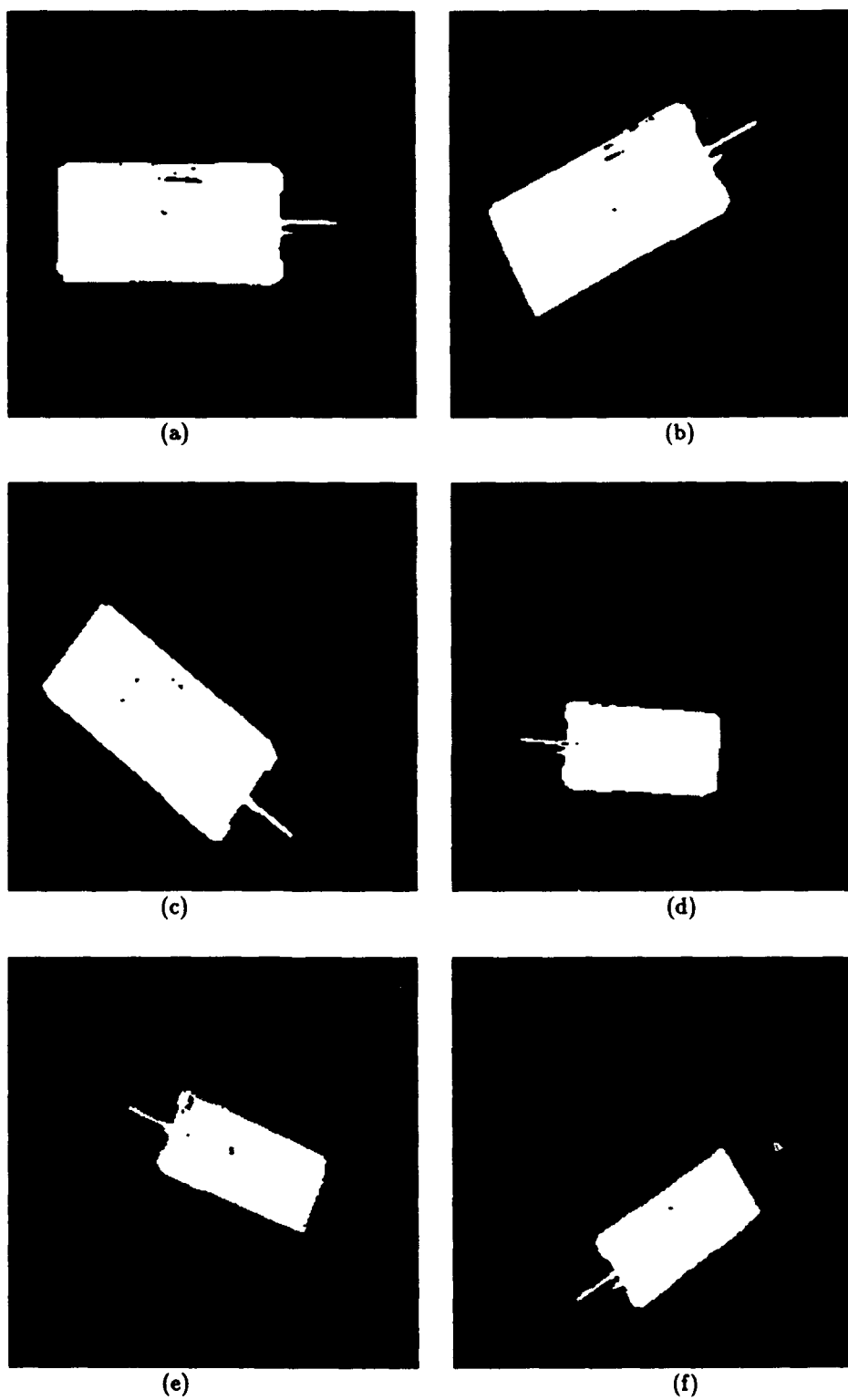
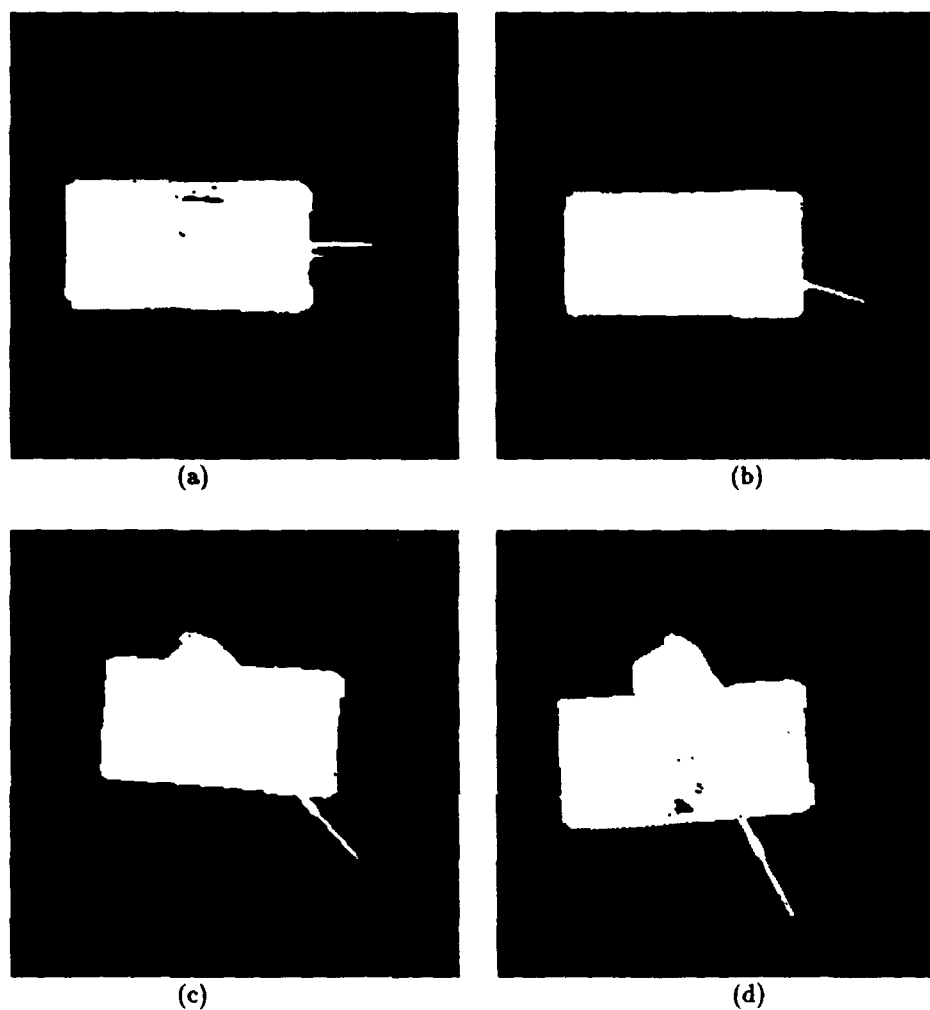


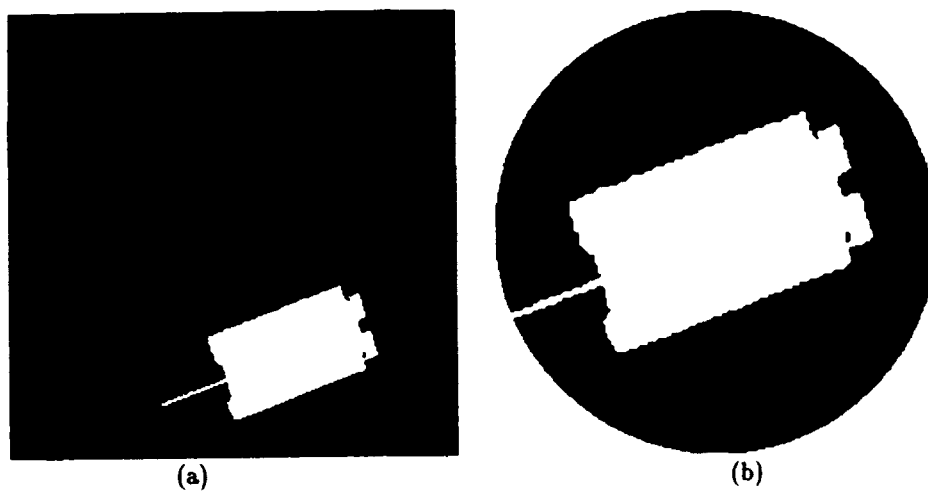
Image 4.2

Database example of the relative translation, scaling and rotation of the projection of an object onto the image plane.





**Image 4.3** Database example of images of one and the same object with different turret positions resulting in quite different projections onto the image plane.



**Image 5.1** Translation and scaling of the object in the image plane: (a) original image; (b) centred and scaled object (unit circle superimposed);

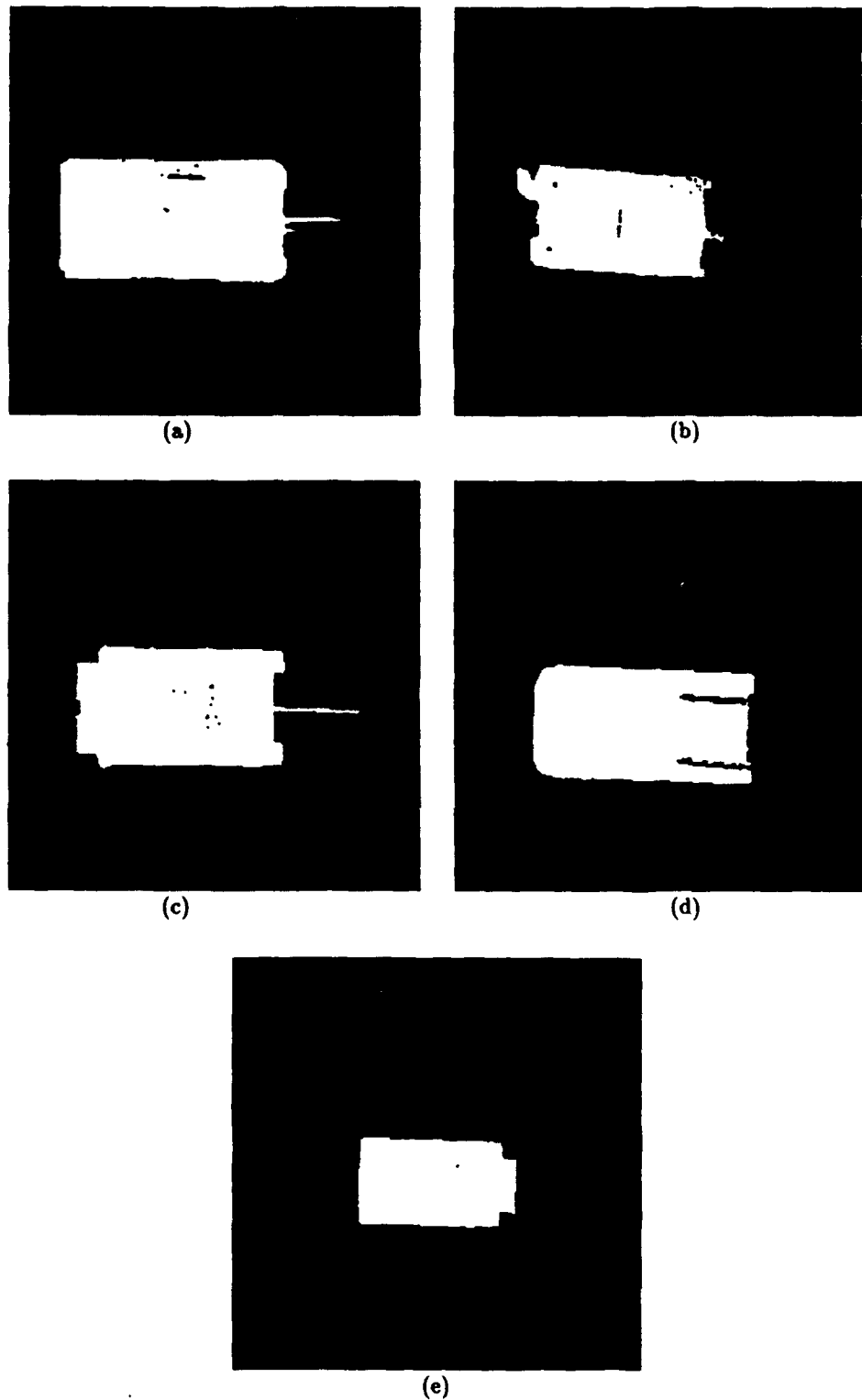


Image 5.2

Database images: topview onto five different objects: (a) Leopard 2 tank; (b) M109 Howitzer; (c) T-80 MBT; (d) Gepard; (e) M113 APC;

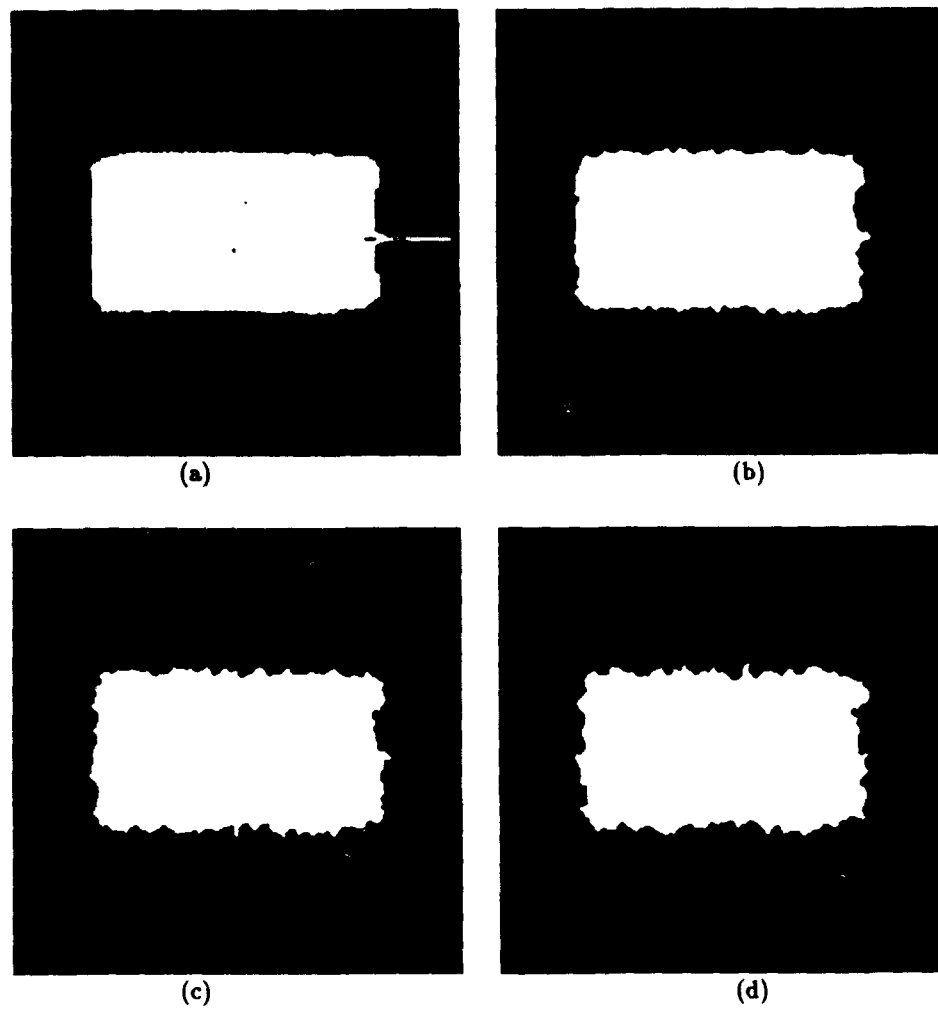


Image 5.3 Influence of the image noise on the object segmentation: (a) original image; (b) SNR 36 dB; (c) SNR 25 dB; (d) SNR 20 dB;

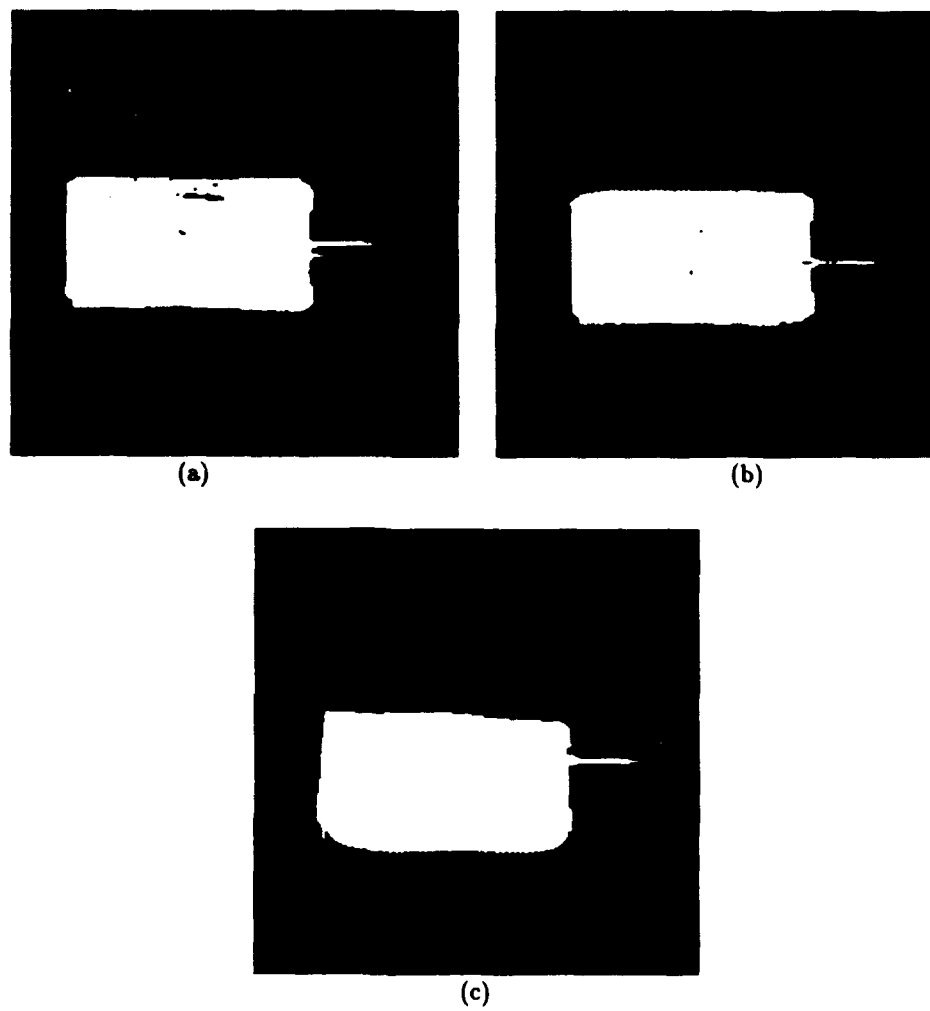


Image 5.4 Database example of images of one and the same object viewed from different viewing directions  $\alpha$ : (a) topview,  $\alpha=90^\circ$ ; (b)  $\alpha=60^\circ$ ; (c)  $\alpha=30^\circ$ ;

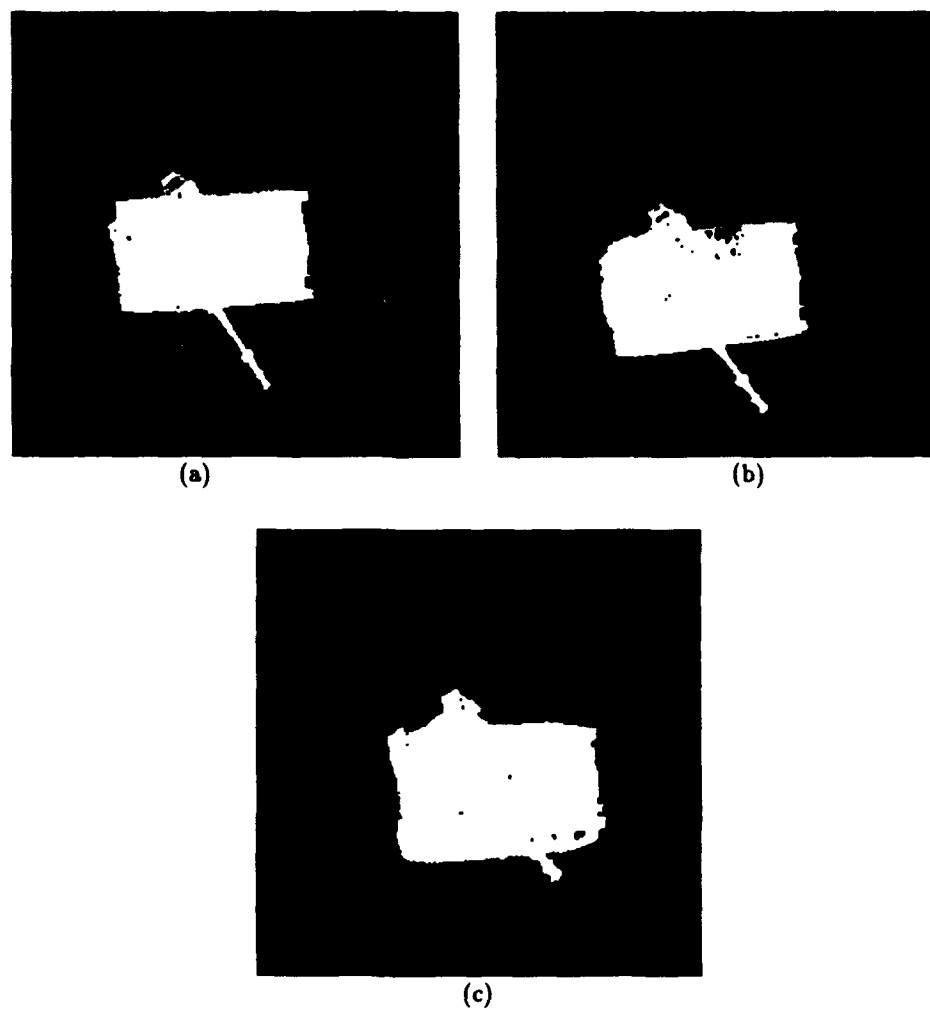


Image 5.5 Database example of images of one and the same object viewed from different viewing directions  $\alpha$ : (a) topview,  $\alpha=90^0$ ; (b)  $\alpha=60^0$ ; (c)  $\alpha=30^0$ ;

## Appendix A

## APPENDIX A TRAINING NEURAL NETWORKS

## Error function

Training a neural network with a set of training vectors comprising examples of all classes under consideration is equivalent to determining an optimal set of interconnection weights. This optimal set is found making use of an iterative process. During each iteration step the weights are changed or adapted so as to implement a gradient descent on an error function  $E()$ . This error function is the mean squared error obtained by the network over the entire training set of input patterns as given in Equation A.1

$$E = \sum_{p=1}^{N_p} \left( 1/2 \sum_j (t_{pj} - o_{pj})^2 \right) \quad (\text{A.1})$$

Here  $N_p$  is the number of patterns in the training set,  $t_{pj}$  the target output value for the  $j$ th component of the output pattern for pattern  $p$  and  $o_{pj}$  the  $j$ th element of the actual network output as a result of the presentation of input pattern  $p$ . Since in practice the weights are adapted after each pattern is presented, the learning algorithm departs in some extent from a true gradient decent in the error function. But with a small enough learning rate the function still will be minimized.

## Adaptation of the Weight Factors

From the network response to a training input pattern and the desired network output an error signal can be determined for each output node. This error signal is used to adjust the weights of the incoming connections to the nodes.

The derivation of the adaptation or learning rules are omitted. A detailed description is given in [Rumelhart et al., '86]. We will suffice by giving the overall results. The basic equations representing the error signal  $\delta_{oi}$  and learning rule for a node on the output layer is given in Equation A.2

$$\delta_{oi} = f'(s_i)(t_i - o_{oi}) \quad (\text{A.2a})$$

## Appendix A

$$\Delta w_{ij}(n+1) = \alpha_o \delta_{oi} o_{hj} + \beta_o \Delta w_{ij}(n) \quad (\text{A.2b})$$

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n+1) \quad (\text{A.2c})$$

where  $\Delta w_{ij}$  represents the change of the weight factor  $w_{ij}$ ,  $o_{oi}$  the output value of processing element  $i$  in the output layer,  $t_i$  the desired output (target),  $\alpha_o$  the learning rate of the output layer and  $f(s_i)$  the derivative of the activation function i.e.

$$f'(s_i) = o_i(1 - o_i) \quad \text{if sigmoidal activation is used.}$$

As can be seen from Equation A.2 the change of weight  $\Delta w_{ij}$  depends on the error signal  $\delta_{oi}$  of output node  $i$ , the output value  $o_{hj}$  of hidden node  $j$  and a constant learning rate  $\alpha_o$ . A momentum term  $\beta_o$  is introduced to achieve a certain conservatism in the direction the weight factors are adapted. In general the relations  $\alpha_o > 0$  and  $0 < \beta_o < 1$  are valid for the learning rate and momentum term. Optimum values for both  $\alpha_o$  and  $\beta_o$  are problem specific and can only be found by trial and error. Remark: The same learning rule is applied to  $w_{iTH}$ .

It is important to realise that only for the nodes in the output layer a straight forward error signal can be obtained. An error signal for the nodes in the hidden layer, necessary to update the weights between the input layer and the hidden layer is therefore in one sense artificial. However it can be shown that the error signal definition  $\delta_{hi}$  and learning rule for the hidden nodes as given in Equation A.3 indeed decrease the overall network error.

$$\delta_{hi} = [o_{hi}(1 - o_{hi})] \sum_{k=1}^{N_o} \delta_{ok} w_{ki} \quad (\text{A.3a})$$

$$\Delta w_{ij}(n+1) = \alpha_h \delta_{hi} o_{oj} + \beta_h \Delta w_{ij}(n) \quad (\text{A.3b})$$

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n+1) \quad (\text{A.3c})$$

Here  $\delta_{hi}$  represents the error signal of hidden node  $i$ ,  $o_{hi}$  the output value of hidden node  $i$ ,  $\delta_{ok}$  the error signal of output node  $k$ ,  $N_o$  the number of nodes in the output layer and  $w_{ki}$  the weight between hidden node  $i$  and output node  $k$ .



---

**Appendix B****APPENDIX B ACTIVE VISION USER COMMANDS**

- **name**                    `norm_image` - Active Vision image object size converter
  
- **synopsis**                `norm_image input.vff output.vff area [-v]`
  
- **description**            `norm_image` normalizes a bi-valued (logical) image with respect to the number of non-zero pixels in the image. The input image must be either an `avBYTE` image with logical values 0 and 255 or an `avFLOAT` image with logical values 0.0 and 1.0.  
  
The desired number of non-zero pixels in the output image is given by `<area>`.  
  
The command `norm_image` does its work silently. With the `-v` (Verbose) option given, the centre of gravity of the input image, the scale factor, the area of the input image, the desired area and the area of the output image are given.  
  
Due to interpolation and thresholding, the desired area and the area of the output image may slightly differ.
  
- **options**                `-v` Verbose option
  
- **example**                Convert a binary object in an image comprising an arbitrary number of pixels into an object of 12000 pixels:  
  
`example% norm_image object.vff normed_object.vff 12000 -v`
  
- **see also**                -
  
- **remarks**                based on Active Vision 0.1 function calls

---

**Appendix B**

- **name**                    `zernike_template` - Active Vision Zernike template images generator
  
- **synopsis**                `zernike_template input.vff dst order_min order_max [-v]`
  
- **description**            `zernike_template` generates template images in the .VFF file format of Zernike polynomials of order `<order_min>` up to `<order_max>`. The range of the valid moment order parameters lies between 0 and 20. For each `<order><repetition>` combination a separate template is generated and stored in the file `<dst><order><repetition>.vff>`.  
  
The template exists of three bands. One band to calculate the real part of the Zernike moment, one band to calculate the imaginary part of the Zernike moment and one band to calculate the absolute value of the Zernike moment.  
  
The image `<src>` is only used to determine the size of the template images.  
  
The command `zernike_template` does its work silently. With the `-v` (Verbose) option given, the generated template filenames are echoed on screen, during execution.
  
- **options**                `-v` Verbose option
  
- **example**                Obtain the zernike moment template images from order 0 up to order 20 of the same size as the reference image `object.vff` and store the results in the file(s) `moment<m><n>.vff`:  
  
`example% zernike_template object.vff moment 0 20 -v`
  
- **see also**                -
  
- **remarks**                based on Active Vision 0.1 function calls

---

**Appendix B**

- **name**                    `zernike_moment` - Active Vision Zernike complex moments generator
  
- **synopsis**                `zernike_moment input.vff dst order_min order_max [-v]`
  
- **description**            `zernike_moment` generates the Zernike moments of the input image `input.vff` and stores the results in the output file `<dst>`. For each moment, the real, imaginary and absolute moment value are successively stored.  
  
The moments are calculated from order `<order_min>` up to order `<order_max>` for all valid repetition values. The range of the valid moment order parameters lies between 0 and 20.  
  
The command `zernike_moment` does its work silently. With the `-v` (Verbose) option given, the calculated moment values are echoed on screen, during execution.
  
- **options**                `-v` Verbose option
  
- **example**                Obtain complex Zernike moments from order 0 up to order 12 of the input image `object.vff` and store the results in `object.zm`:  
  
`example% zernike_moment object.vff object.zm 0 12 -v`
  
- **see also**                `zernike_fmoment`
  
- **remarks**                based on Active Vision 0.1 function calls

---

**Appendix B**

- **name**                    `zernike_fmoment` - Active Vision fast Zernike complex moments generator
- **synopsis**                `zernike_fmoment input.vff dst template order_min order_max [-v]`
- **description**            `zernike_fmoment` generates the Zernike moments of the input image `input.vff` and stores the results in the output file `<dst>`. For each moment, the real, imaginary and absolute moment value are successively stored. The calculation is based on the predefined Zernike moment templates
- `<<template><order><repetition>.vff>`
- for fast calculation.
- The moments are calculated from order `<order_min>` up to order `<order_max>` for all valid repetition values. The range of the valid moment order parameters lies between 0 and 20.
- The command `zernike_fmoment` does its work silently. With the `-v` (Verbose) option given, the calculated moment values are echoed on screen, during execution.
- **options**                `-v` Verbose option
- **example**                Obtain complex Zernike moments from order 0 up to order 12 of the input image `object.vff` making use of the predefined Zernike templates `<moment>` and store the results in `object.zm`:
- `example% zernike_fmoment object.vff object.zm moment 0 12 -v`
- **see also**                `zernike_moment`, `zernike_template`
- **remarks**                based on Active Vision 0.1 function calls

---

**Appendix B**

- **name**                    `zernike_reconstruct` - Active Vision image reconstructor based on Zernike moments
  
- **synopsis**                `zernike_reconstruct input.vff output.vff momentfile order_min order_max [-v]`
  
- **description**            `zernike_reconstruct` generates an image based on the complex Zernike moments stored in file `<<momentfile><order><repetition>.vff>` from order `<order_min>` up to order `<order_max>`. The range of the valid moment order parameters lies between 0 and 20.  
The output image `<output.vff>` is the sum of the input image `<input.vff>` and the generated image.  
  
The command `zernike_reconstruct` does its work silently. With the `-v` (Verbose) option given, the order and repetition of the moment filenames are echoed on screen, during execution.
  
- **options**                `-v` Verbose option
  
- **example**                Reconstruct an image out of the complex Zernike moments stored in the file `object.zm` from order 6 up to order 12 and add the result to the partially reconstructed image `rec00_05.vff` and store the result in `rec00_12.vff`  
  
`example% zernike_reconstruct rec00_05.vff rec00_12.vff object.zm 6 12 -v`
  
- **see also**                `zernike_moment`
  
- **remarks**                based on Active Vision 0.1 function calls

---

Appendix B

■ **name**                    **zernike\_freconstruct** - Active Vision fast image reconstructor based on Zernike moments

■ **synopsis**                **zernike\_freconstruct input.vff output.vff momentfile templatefile  
                                 order\_min order\_max [-v]**

■ **description**            **zernike\_freconstruct** generates an image based on the complex Zernike moments stored in file <<momentfile><order><repetition>.vff> from order <order\_min> up to order <order\_max>. The range of the valid moment order parameters lies between 0 and 20.  
The output image <output.vff> is the sum of the input image <input.vff> and the generated image.

The calculation is based on the predefined Zernike moment templates

<<templatefile><order><repetition>.vff>

for fast calculation.

The command **zernike\_freconstruct** does its work silently. With the -v (Verbose) option given, the order and repetition of the moment filenames are echoed on screen, during execution.

■ **options**                -v Verbose option

■ **example**                Reconstruct an image out of the complex Zernike moments stored in the file object.zm from order 6 up to order 12 and add the result to the partially reconstructed image rec00\_05.vff and store the result in rec00\_12.vff:

**example%**                **zernike\_freconstruct rec00\_05.vff rec00\_12.vff object.zm  
                                 moment 6 12 -v**

■ **see also**                **zernike\_fmoment**

■ **remarks**                based on Active Vision 0.1 function calls

---

**Appendix B**

- **name** `nll_edge` - Active Vision edge detector based on Nonlinear Laplace operator
  
- **synopsis** `nll_edge input.vff output.vff -g [0|3|5] -l [3|5] [-v]`
  
- **description** `nll_edge` generates an edge image of the input image `<input.vff>` and stores the result in `<output.vff>`. The input image can be preprocessed by a Gauss kernel by selecting `<-g 3>` ( $\sigma = \sqrt{0.5}$ ) or `<-g 5>` ( $\sigma = \sqrt{2.0}$ ). For `<-g 0>` no Gauss kernel is applied. The Nonlinear Laplace kernel size can be chosen between `<-l 3>` and `<-l 5>`.  
  
The command `nll_edge` does its work silently. With the `-v` (Verbose) option given, the different processing steps are echoed on screen, during execution.  
  
By thresholding the output image a bi-valued edge image can be obtained.
  
- **options** `-v` Verbose option
  
- **example** Obtain avFLOAT edge image `<output.vff>` of input image `<input.vff>`, preprocessing the input image with a Gauss kernel of size 5, by applying a Nonlinear Laplace operator of size 5:  
  
`example% nll_edge input.vff output.vff -g 5 -l 5 -v`
  
- **see also** Vliet, L.J. v., and Young, I.T., "A Nonlinear Laplace Operator as Edge Detector in Noisy Images", Computer Vision, Graphics, and Image Processing, 45, pp. 167-195, 1989.
  
- **remarks** based on Active Vision 0.1 function calls

## REPORT DOCUMENTATION PAGE

(MOD-NL)

1. DEFENSE REPORT NUMBER (MOD-NL) T 92-3840	2. RECIPIENT'S ACCESSION NUMBER	3. PERFORMING ORGANIZATION REPORT NUMBER FEL-92-A394
4. PROJECT/TASK/WORK UNIT NO. 22974	5. CONTRACT NUMBER A92KL632	6. REPORT DATE DECEMBER 1992
7. NUMBER OF PAGES 87 (INCL. 2 APPENDICES, EXCL. RDP & DISTRIBUTION LIST)	8. NUMBER OF REFERENCES 39	9. TYPE OF REPORT AND DATES COVERED FINAL
10. TITLE AND SUBTITLE ZERNIKE MOMENTS AND ROTATION INVARIANT OBJECT RECOGNITION A NEURAL NETWORK ORIENTED CASE STUDY		
11. AUTHOR(S) P.F.C. KREKEL		
12. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) TNO PHYSICS AND ELECTRONICS LABORATORY, P.O. BOX 96864, 2509 JG THE HAGUE OUDERWAALSDORPERWEG 63, THE HAGUE, THE NETHERLANDS		
13. SPONSORING/MONITORING AGENCY NAME(S) DEBKL/DCAWACO		
14. SUPPLEMENTARY NOTES THE CLASSIFICATION DESIGNATION ONGERUBRICEERD IS EQUIVALENT TO UNCLASSIFIED.		
15. ABSTRACT (MAXIMUM 200 WORDS, 1044 POSITIONS)  IN THIS REPORT THE RESULTS OF THE FEASIBILITY STUDY INVESTIGATING THE CHARACTERISTICS OF COMPLEX ZERNIKE MOMENTS AND THEIR APPLICATION IN TRANSLATION-, SCALE- AND ROTATION-INVARIANT OBJECT RECOGNITION PROBLEMS IS PRESENTED. THE COMPLEX ZERNIKE MOMENTS ARE USED AS CHARACTERISING FEATURES IN A NEURAL NETWORK BASED TARGET RECOGNITION APPROACH FOR THE CLASSIFICATION OF OBJECTS IN IMAGES RECORDED BY SENSORS MOUNTED ON AN AIRBORNE PLATFORM. THE COMPLEX ZERNIKE MOMENTS ARE A TRANSFORMATION OF THE IMAGE BY THE PROJECTION OF THE IMAGE ONTO AN EXTENDED SET OF ORTHOGONAL POLYNOMIALS. THE MAIN EMPHASIS OF THIS STUDY IS LAID ON THE PERFORMANCE EVALUATION OF NEURAL NETWORKS. THEREFORE, THREE TYPES OF CLASSIFIERS ARE EVALUATED: A MULTI-LAYER PERCEPTRON (MLP) NEURAL NETWORK, A BAYES STATISTICAL CLASSIFIER AND A NEAREST-NEIGHBOUR CLASSIFIER. EXPERIMENTS ARE BASED ON A SET OF BINARY IMAGES SIMULATING MILITARY VEHICLES EXTRACTED FROM THE NATURAL BACKGROUND. FROM THESE EXPERIMENTS THE CONCLUSIONS CAN BE DRAWN THAT COMPLEX ZERNIKE MOMENTS ARE EFFICIENT AND EFFECTIVE OBJECT CHARACTERISING FEATURES THAT ARE ROBUST UNDER ROTATION OF THE OBJECT IN THE IMAGE AND TO A CERTAIN EXTENT UNDER VARYING AFFINE PROJECTIONS OF THE OBJECT ONTO THE IMAGE.		
16. DESCRIPTORS NEURAL NETWORKS AUTOMATIC PATTERN RECOGNITION		IDENTIFIERS OBJECT RECOGNITION ZERNIKE MOMENTS
17a. SECURITY CLASSIFICATION (OF REPORT) ONGERUBRICEERD	17b. SECURITY CLASSIFICATION (OF PAGE) ONGERUBRICEERD	17c. SECURITY CLASSIFICATION (OF ABSTRACT) ONGERUBRICEERD
18. DISTRIBUTION/AVAILABILITY STATEMENT  UNLIMITED		17d. SECURITY CLASSIFICATION (OF TITLES) ONGERUBRICEERD